CPC710 PCI Bridge

and Memory Controller

User Manual

SA14-2571-02

Fourth Edition (September 2002)

This edition of the *IBM CPC710 PCI Bridge and Memory Controller User Manual* applies to the IBM CPC710 PCI bridge and memory controller, until otherwise indicated in new versions or application notes.

© Copyright International Business Machines Corporation 2002

All Rights Reserved
Printed in the United States of America September 2002

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both.
IBM IBM Logo
CoreConnect
PowerPC PowerPC logo
PowerPC Architecture
RISCTrace RISCWatch

Other company, product, and service names may be trademarks or service marks of others.

All information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in implantation, life support, space, nuclear, or military applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change IBM product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN “AS IS” BASIS. In no event will IBM be liable for damages arising directly or indirectly from any use of the information contained in this document.

IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6351

The IBM home page can be found at http://www.ibm.com

The IBM Microelectronics Division home page can be found at http://www.ibm.com/chips

Contents

About This Book xiii

Chapter 1. Overview 1-1

Features 1-1
- 60x Bus Interface 1-1
- Memory Controller 1-2
- PCI-32 and PCI64 Bus Bridges 1-2

Chapter 2. Addressing Model 2-1

Address Maps 2-1
CPU to PCI Addressing Model 2-2
- PREP and FPHB Modes 2-2
- CHRP Mode 2-3
- Peripheral I/O Address Translation 2-4

PCI to System Memory 2-5
- PowerPC Reference Platform (PREP) Mode 2-6
  - PCI Master Address Operation 2-6
  - Translation Enabled in PREP Mode 2-6
  - Translation Disabled in PREP Mode 2-7
- Flexible PCI Host Bridge (FPHB) Mode 2-8
- CHRP Address Map 2-9

Chapter 3. System I/O Interface 3-1

Configuration 3-1
System I/O Registers: Application Presence Detect Bits 3-1
Flash Interface 3-1
- Boot Rom 3-1
- Extended Boot Flash 3-2
- Byte ordering in the Boot & Extended Flash for 8 bit bus size: 3-2

Chapter 4. 60x Interface 4-1

Endian Support 4-1
- PowerPC Processor Behavior Mode 4-4
  - Processor Behavior in LE Mode 4-4
  - Endian Behavior 4-4

60x Bus Arbiter Description 4-5
- Rotating Priority Resolution 4-6
- Address Bus Pipelining 4-6
- Arbiter Requirements 4-6
  - Internal ABB 4-6
  - Qualified SYS_BG Equation 4-6
  - SYS_TS Assertion 4-7
  - SYS_BR Negation 4-7
  - Qualified SYS_DBG Equation 4-7
  - High Impedance After SYS_TEA 4-7
  - SYS_DRTRY Assertion 4-7
  - Slave Data Bus Determination 4-7
  - SYS_L2_Hit Assertion 4-7
- Bus Enhancements 4-8
  - DBB not Required by Masters 4-8
  - Half-Cycle Precharge not Required on SYS_TA 4-8
  - SYS_ARTRY_PREV in QDBG Equation Eliminated 4-8
- 60x Bus Transfer Types and Sizes 4-8

Data Gathering 4-10
SYNC and EIEIO 4-11
Address Retry (SYS_ARTRY) 4-12
Precharging SYS_ARTRY and SYS_SHD 4-12
SYS_ARTRY Assertions 4-12
Recommended SYS_ARTRY Procedure 4-12
Deadlock Avoidance 4-13
Deadlock Avoidance Operation with Internal Logic 4-14
Deadlock Avoidance Operation with External Logic 4-15
Error Handling for CPU-Initiated Transactions 4-15
Checkstop Errors 4-15
Chapter 5. Initialization 5-1
CPC710 Power Up Sequence: 5-1
POWERGOOD Power-On Reset 5-1
PLL for Clock System 5-2
Initialization of the SDRAM 5-3
Reset Individual Devices 5-4
Reset in Multiprocessor mode 5-4
Typical Register Setup Sequence 5-5
Chapter 6. Memory Controller 6-1
Overview 6-1
Bank Definitions 6-2
SDRAM Banks 6-2
DIMM Banks 6-2
Interleaved Banks 6-2
Memory Signal Connections 6-3
SDRAM Subsystem Overview 6-6
Supported SDRAM Organizations 6-7
SDRAM Buffering Requirements 6-8
Typical SDRAM Signals 6-8
CKE (Clock Enable) Signal 6-9
Mapping of System address to SDRAM Memory address with the CPC710 6-9
Memory Controller Registers 6-11
SDRAM0_MCCR Register 6-11
SDRAM0_MCERx Registers 6-14
Error Handling 6-15
Single-Bit ECC Error, General Case 6-16
Single-Bit ECC Error, Special Case 6-16
Invalid Address Error 6-16
Double-Bit ECC Error, General Case 6-16
Double-Bit ECC Error, Special Case 6-17
Overlapping Memory Extents 6-17
ECC Check-bit and Syndrome 6-17
Single-Bit Error Correction 6-18
Additionnal Information for Software 6-19
Chapter 7. PCI Bridges 7-1
Address Map 7-1
System Standard Configuration Registers 7-1
System PHB Registers 7-2
PCI Bus Commands 7-2
PCI Master Memory Read Cycles 7-2
PCI Master Memory Write Cycles 7-3
Configuration Cycles 7-4
Type 0 Configuration Cycles 7-6

Type 1 Configuration Cycles 7-6
PCI Performance Estimates 7-6
PCI Master Error Handling 7-7
Chapter 8. DMA Controller 8-1
Introduction 8-1
Mode of operation of the DMA 8-1
Starting the DMA 8-1
DMA Transfer Registers 8-2
DMA Transfer Status Cache Line Descriptor for Chained DMA's 8-3
DMA Procedure 8-4
Special Boundary Conditions 8-5
Chapter 9. Register Summary 9-1
System Register Space 9-1
Standard PCI Configuration Space 9-5
Standard PCI Configuration Registers 9-6
Specific PCI Host Bridge Registers 9-8
Alphabetical List of Registers 9-9
Chip Control Registers 9-10
CPC0_ABCNTL 9-10
CPC0_ATAS 9-13
CPC0_AVDG 9-15
CPC0_ERRC 9-17
CPC0_GPDIR 9-19
CPC0_GPIN 9-20
CPC0_GPOUT 9-21
CPC0_MPSR 9-22
CPC0_PCIBAR 9-23
CPC0_PCICNFR 9-24
CPC0_PCIENB 9-25
CPC0_PGCHP 9-26
CPC0_PIDR 9-29
CPC0_RGBAN0 9-30
CPC0_RGBAN1 9-31
CPC0_RSTR 9-32
CPC0_RTBR 9-33
CPC0_SEAR 9-34
CPC0_SESR 9-35
CPC0_SIOC0 9-38
CPC0_SIOC1 9-40
CPC0_SPOR 9-41
CPC0_SRST 9-42
CPC0_UCTL 9-43
DMA Registers 9-45
DMA0_GSCRP, DMA0_GSCRU 9-45
DMA0_XCLRP, DMA0_XCLRU 9-47
DMA0_XPARP, DMA0_XPARU 9-48
DMA0_XSCRP, DMA0_XSCRU 9-49
DMA0_XSSRP, DMA0_XSSRU 9-50
DMA0_XTARP, DMA0_XTARU 9-52
DMA0_XWARP, DMA0_XWARU 9-53
PCI Configuration Registers 9-54
PCIC0_DLKRETRY 9-54
PCIC1_INTRESET 9-55

PCIC1_ITADDSET 9-56
PCIC1_PPBAR 9-57
PCIC1_PSBAR 9-58
PCICx_BIST 9-59
PCICx_BUSNO 9-60
PCICx_CACHELS 9-61
PCICx_CLS 9-62
PCICx_CMD 9-63
PCICx_DEVID 9-65
PCICx_DISCNT 9-66
PCICx_HDTYPE 9-67
PCICx_INTLN 9-68
PCICx_INTPN 9-69
PCICx_LATTIM 9-70
PCICx_MAXLTNCY 9-71
PCICx_MINGNT 9-72
PCICx_RETRY 9-73
PCICx_REVID 9-74
PCICx_STATUS 9-75
PCICx_SUBNO 9-77
PCICx_VENDID 9-78
PCI Local Registers 9-79
PCIL0_PPBAR 9-79
PCIL0_PSBAR 9-80
PCIL1_INTSET 9-81
PCIL1_ITADDRESET 9-82
PCILx_ACR 9-83
PCILx_BARPP 9-84
PCILx_BARPS 9-85
PCILx_BIODLK 9-86
PCILx_BPMDLK 9-87
PCILx_CFGADDR 9-88
PCILx_CFGDATA 9-89
PCILx_CRR 9-90
PCILx_CSR 9-91
PCILx_CTRLW 9-92
PCILx_DLKCTRL 9-93
PCILx_DLKDEV 9-95
PCILx_INTACK 9-96
PCILx_IOSIZE 9-97
PCILx_MSIZE 9-98
PCILx_PCIDG 9-99
PCILx_PIBAR 9-100
PCILx_PLSSR 9-101
PCILx_PMBAR 9-102
PCILx_PPSIZE 9-103
PCILx_PR 9-104
PCILx_PSEA 9-106
PCILx_PSRCR 9-107
PCILx_PSSIZE 9-109
PCILx_PSWCR 9-110
PCILx_SIBAR 9-112
PCILx_SMBAR 9-113
PCILx_TIODLK 9-114

PCILx_TPMDLK 9-115
Memory Controller Registers 9-116
SDRAM0_MCCR 9-116
SDRAM0_MCER0:5 9-120
SDRAM0_MEAR 9-123
SDRAM0_MESR 9-124
SDRAM0_MWPR 9-125
SDRAM0_SIOR0 9-127
SDRAM0_SIOR1 9-128
Chapter 10. Timing Diagrams 10-1
CPU to Memory Transactions 10-1
CPU Access to the Boot ROM 10-6
PCI64 External Master Accessing SDRAM Memory 10-7
Chapter 11. Signal Summary 11-1
Index X-1
Revision Log R-1

Figures

Figure 1-1. System Block Diagram 1-3
Figure 1-2. Component Block Diagram 1-4
Figure 1-3. Internal Buffering and Data Flow 1-5
Figure 2-1. Memory Map 2-1
Figure 2-2. CPU to PCI Addressing Model (PREP and FPHB Modes) 2-2
Figure 2-3. CPU to PCI Addressing Model (CHRP Mode) 2-3
Figure 2-4. Noncontiguous I/O Address Mode Enabled 2-5
Figure 2-5. Address Translation Enabled in PREP Mode 2-7
Figure 2-6. Address Translation Disabled in PREP Mode 2-7
Figure 2-7. PCI to System Addressing Model (FPHB Mode) 2-8
Figure 2-8. CHRP Address Map 2-9
Figure 3-1. Connection of Boot ROM and System I/O Registers to Device 3-3
Figure 4-1. CPC710 Endian Logic 4-2
Figure 4-2. Processor Data Bus Byte Swap for Little Endian 4-5
Figure 4-3. Data Gathering Algorithm 4-11
Figure 4-4. Deadlock avoidance circuits in the CPC710 4-14
Figure 5-1. Power Up Sequence 5-1
Figure 5-2. PLL Reset 5-3
Figure 5-3. Arbitration to support of 4 Way Multiprocessing with the CPC710 5-5
Figure 6-1. DIMM Bank Configuration 6-2
Figure 6-2. Programming with Single Bank DIMMs 6-3
Figure 6-3. Programming with Dual Bank DIMMs 6-3
Figure 6-4. SDRAM Interface Block Diagram 6-6
Figure 6-5. SDRAM Commands issued by the CPC710 6-9
Figure 6-6. Use of the CKE signal for SDRAM Control 6-9
Figure 7-1. PCI Memory Read State Diagram 7-3
Figure 7-2. PCI Memory Write State Diagram 7-4
Figure 7-3. PCI Address/Data Bus for Type 0 Configuration Cycles 7-6
Figure 9-1. CPC710 Register Address Map 9-2
Figure 9-2. PCI Configuration Space 9-6
Figure 10-1. Read Page Hit from PowerPC CPU to SDRAM 10-1
Figure 10-2. Read Page Miss from PowerPC CPU to SDRAM 10-2
Figure 10-3. Write Burst Page Hit from PowerPC CPU to SDRAM 10-3
Figure 10-4. Write Burst Page Miss from PowerPC CPU to SDRAM 10-4
Figure 10-5. Write One Byte to Memory from CPU: Read Modify Write 10-5
Figure 10-6. Read of One Byte from the Boot ROM 10-6
Figure 10-7. Write of One Byte to the Boot Flash 10-6
Figure 10-8. Read 32 Bytes from SDRAM by a PCI Master on a 66MHz PCI64 bus: 10-7
Figure 10-9. Write 32 Bytes to SDRAM from a PCI Master on the 66MHz PCI64 bus 10-8

Tables

Table 4-1. Processor Little Endian Address Modification 4-4
Table 4-2. Processor Little Endian Address Unmunge Equations 4-5
Table 4-3. Non-Burst Transactions (SYS_TBST = 1) 4-8
Table 4-4. Burst Transactions (SYS_TBST = 0) 4-9
Table 4-5. Transfer Types (Page 1 of 2) 4-9
Table 4-6. CPU Initiated Transactions (Page 1 of 5) 4-15
Table 5-1. PLL Inputs Control Signal Setting 5-2
Table 6-1. Memory Performance for Cache Line Operations (ECC Active) 6-1
Table 6-2. SDRAM Common Signals 6-3
Table 6-3. External MUX Controller for Memory Data 6-4
Table 6-4. Memory Address Bit Definition for Non-Row Column Addressing Bits 6-4
Table 6-5. SDRAM Subsystem Signals 6-4
Table 6-6. SDRAM DIMM Chip Select Connections Example 6-4
Table 6-7. Supported DIMMs 6-7
Table 6-8. SDRAM Input Signal Frequencies 6-8
Table 6-9. System Address Mapping 6-10
Table 6-10. SDRAM Address Mapping 6-10
Table 6-11. SDRAM Control Register Programming 6-11
Table 6-12. SDRAM0_MCERx to Program Functions of DIMMs 6-14
Table 6-13. SDRAM0_MCERx Register Initialization 6-15
Table 6-14. ECC Check-Bit /Single-Bit Error Syndrome Matrix 6-18
Table 6-15. Data Values Required for Check Bits 6-19
Table 7-1. PCI32 Bus Device Physical Connection Example 7-1
Table 7-2. PCI Bus Bridge Configuration Address Map 7-1
Table 7-3. Supported PCI Commands 7-2
Table 7-4. PCI Configuration Cycle Matrix 7-5
Table 7-5. PCI to Memory Sustained Throughput 7-6
Table 7-6. CPU to PCI Sustained Throughput 7-7
Table 7-7. PCI Master Error Handling (Page 1 of 2) 7-8
Table 8-1. DMA Transfer Register Summary 8-2
Table 8-2. DMA Transfer Status Cache Line Definition 8-3
Table 9-1. System Registers List 9-3
Table 9-2. Standard PCI Configuration Registers 9-7
Table 9-3. Specific PCI Host Bridge Registers 9-8

About This Book

This book describes the IBM CPC710 PCI bridge and memory controller, a highly integrated host bridge device that interfaces a PowerPC 60x bus with SDRAM-based system memory and two PCI ports. One PCI bridge supports a standard 32-bit, 33MHz PCI bus for standard and native I/O. The other PCI bridge supports a 64-bit, 33-66MHz PCI bus for high data throughput applications such as graphics and high-speed communications.

Who Should Use This Book

This book is for system hardware and software developers. The audience should understand embedded processor design, embedded system design, operating systems, RISC processing, and design for testability.

How to Use This Book

This book contains the following chapters:

“Contents”

“Figures”

“Tables”

Chapter 1, “Overview”

Chapter 2, “Addressing Model”Chapter 8, “DMA Controller”

Chapter 3, “System I/O Interface”

Chapter 4, “60x Interface”

Chapter 5, “Initialization”

Chapter 6, “Memory Controller”

Chapter 7, “PCI Bridges”

Chapter 8, “DMA Controller”

Chapter 9, “Register Summary”

Chapter 10, “Timing Diagrams”

Chapter 11, “Signal Summary”

“Index”

“Revision Log”

Conventions and Notation

The use of overbars, for example $\overline{\text{RESET}}$, designates signals that are active low. All signals are active high unless shown with an overbar.

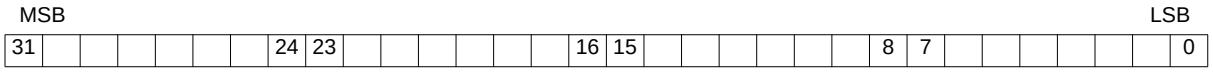
Decimal, hexadecimal, and binary numbers are used throughout this document, and are labeled as follows:

Decimal:	1234.56
Hexadecimal:	x'ABCD'
Binary:	b'0101'

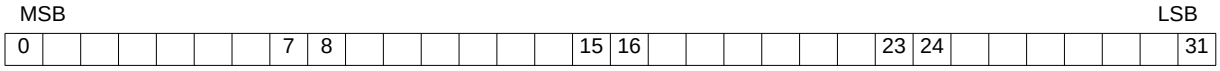
Example

Hexa	Binary
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

In **Little Endian** mode, bits and bytes are numbered in descending order from left to right. The most significant bit (MSB) has the highest number and the least significant bit (LSB) has the lowest number:



In **Big Endian** mode, bits and bytes are numbered in ascending order from left to right. The most significant bit (MSB) has the lowest number and the least significant bit (LSB) has the highest number:



Related Publications

The following publications contain related information:

CPC710 PCI Bridge and Memory Controller Data Sheet, SA14-2572-00

IBM PowerPC 750CX/750CXe RISC Microprocessor User's Manual

CPC700 Memory Controller and PCI Bridge Data Sheet

CPC700 Memory Controller and PCI Bridge User's Manual

Chapter 1. Overview

The CPC710-133 is a highly integrated host bridge device that interfaces a PowerPC 60x bus with SDRAM-based system memory and two PCI ports. It provides arbitration for one up to four processors and supports two levels of pipelining per processor along with 64-byte buffers.

The CPC710-133 memory controller supports SDRAM, allowing the memory to burst data on almost every bus cycle at 100 or 133 MHz (1-1-2-1 after initial latency on Read and 1-1-1-1 on write).

For system designs requiring high I/O bandwidth, the device contains two PCI host bus bridges. One bridge supports a standard 32-bit, 33 MHz PCI bus for standard and native I/O. The other bridge supports a 64-bit, 33-66 MHz PCI bus for high data throughput applications such as graphics and high-speed communications.

A DMA controller provides high speed capability for large data transfers between memory and I/O. Store-gathering enhances CPU-to-I/O performance.

1.1 Features

- Up to 133 MHz PowerPC 60x 64-bit bus
- 2.5 volts 60X PowerPC bus
- Supports 100 and 133 MHz SDRAM including PC100 and PC133
- Up to 2 MB flash Boot ROM support
- 32-bit 33 MHz/64-bit 33-66 MHz async dual bus
- Reads two external 32-bit registers
- PreP and CHRP compliant design
- One-channel chained DMA controller
- Up to 256 MB Extended Flash support
- 3.3 volts $\pm$ 5%, 2.5 volts $\pm$ 2.5%
- -40 to 85°C junction temperature
- Power dissipation 2.1 watts typical at 3.3 volts - 100 MHz.
- FC-PBGA package, 729 pins, 1.27mm pitch, 35x35mm
- CMOS SA-12E, 0.25µm technology
- PLL to reduce on-chip system clock skew
- JTAG controller (LSSD design)

1.1.1 60x Bus Interface

- Supports PowerPC 750L, 750CX, 750CXe processors
- up 133 MHz external bus operation
- Supports four processors or L2 lookaside cache
- Dual 32-byte store back buffers
- High bandwidth 2-way arbiter
- Little Endian mode PowerPC

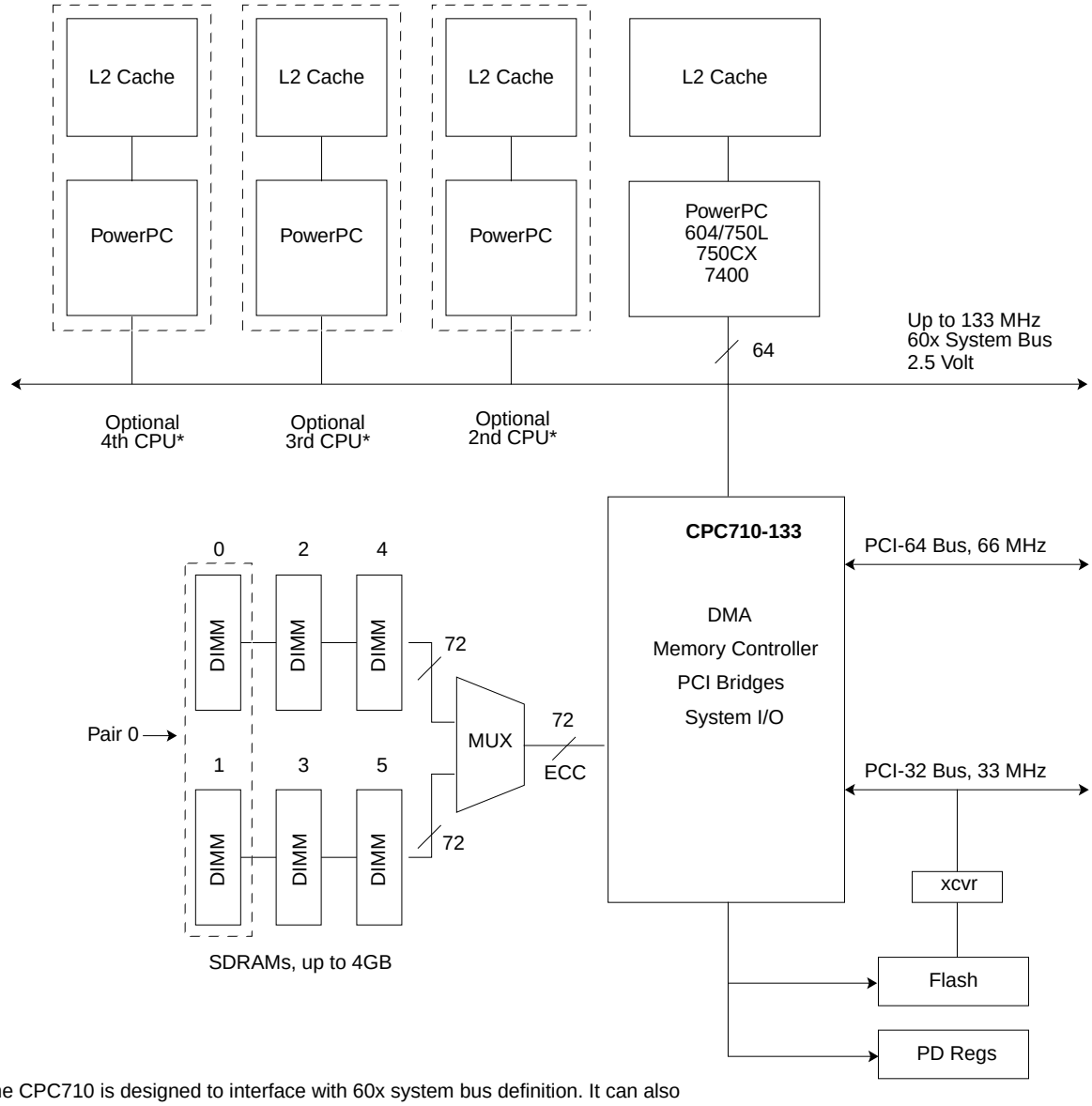
- Supports SYNC/EIEIO ordering operations
- Supports 60x bus configuration cycles

1.1.2 Memory Controller

- Supports 100 and 133 MHz SDRAM including PC100 and PC133 and Registered SDRAM
- Up to 4GB
- 2-way interleaved SDRAM with ECC (external MUX to reduce pin count)
- Supports 16, 64, 128, and 256 Mb SDRAMs
- Programmable timing parameters
- Up to 6 dual bank DIMM
- Up to 4 banks supported for Multibanking
- SDRAM Access command queue with look ahead override option for CPU, PCI's, and DMA
- Access based on 32-byte cache line reload
- Three separate dual 32-byte load buffers (PCI-32, PCI64, 60x)

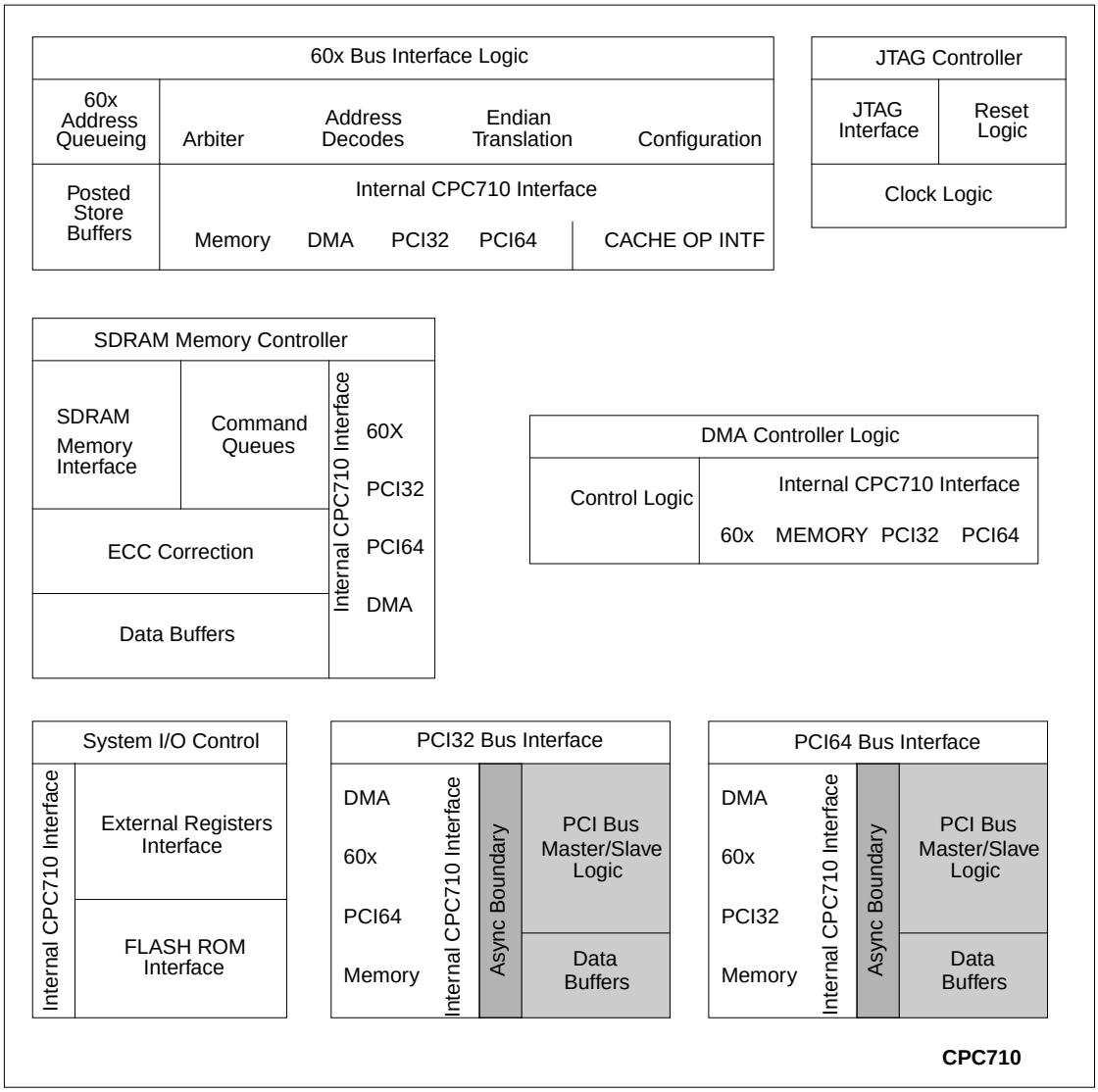
1.1.3 PCI-32 and PCI64 Bus Bridges

- Two independent PCI bus bridges with parking
- PCI revision 2.1 compliant
- PCI32 3.3V Compliant with 5.0 V PCI signalling
- PCI64 3.3V
- Runs async logic to 60x and memory controller
- PCI64 arbitration can be disabled
- Dual 32-byte buffers in each PCI bus bridge
- Round-robin PCI arbiter
- Coherency for memory access through DMA controller or through PCI master.
- Noncontiguous byte enable transfer to memory
- The CPC710 is single load on all PCI signals



*The CPC710 is designed to interface with 60x system bus definition. It can also directly interface to 1–4 PowerPC 750/7400 processors.

Figure 1-1. System Block Diagram



Clocked at System Bus Speed

Clocked at PCI32 Bus Speed (33 MHz)

Clocked at PCI64 Bus Speed (33-66 MHz)

Dual Clocked Logic

Figure 1-2. Component Block Diagram

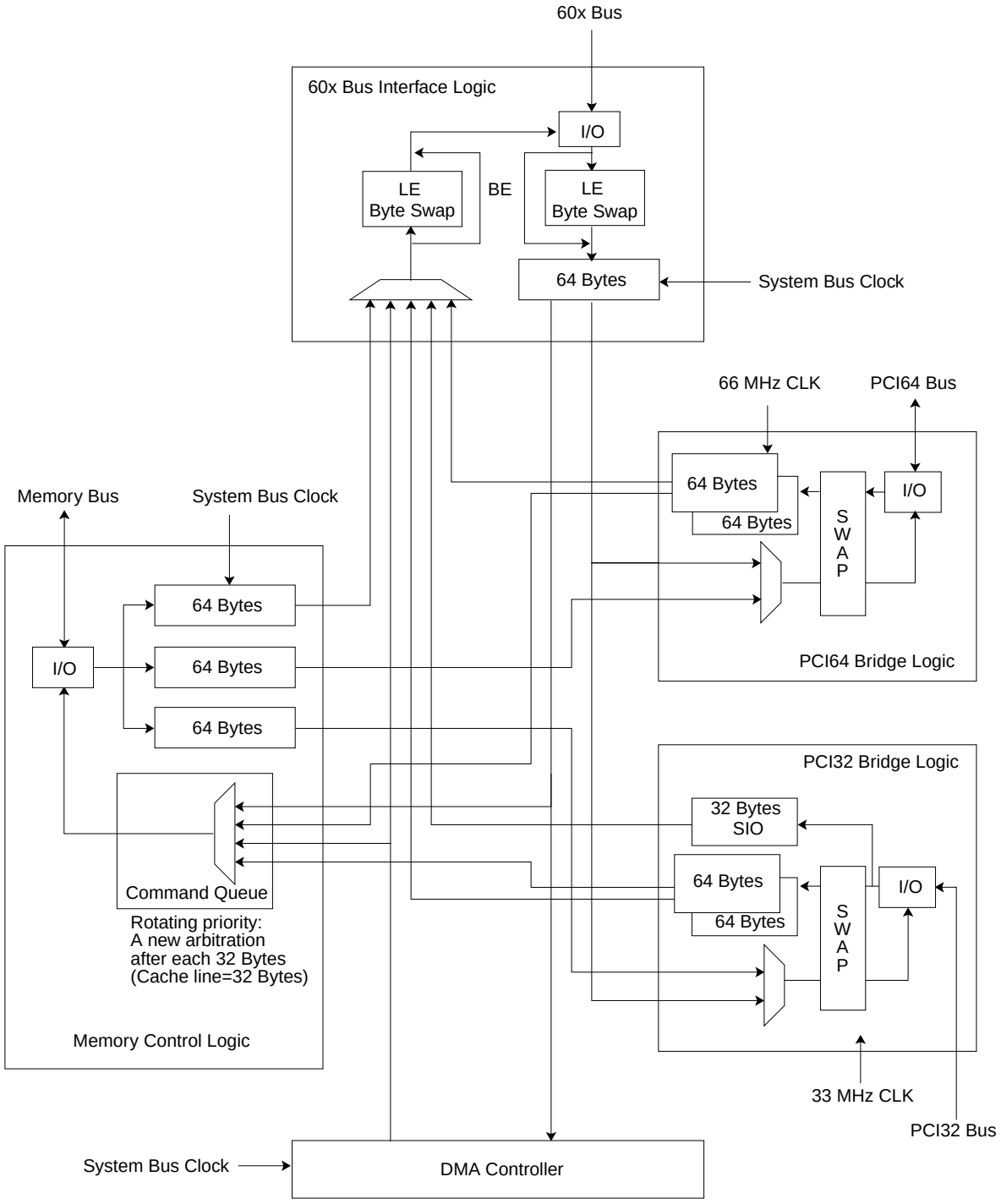


Figure 1-3. Internal Buffering and Data Flow

Chapter 2. Addressing Model

2.1 Address Maps

The CPC710 address spaces can be programmed in 3 different modes

- PREP: PowerPC REference Platform based on PREP specification
- CHRP: Common Hardware Reference Platform based on CHRP specification.
- FPHB: Flexible PCI Host Bridge Mode.

The FPHB Mode provides an address map that is highly programmable and allows for configuration of select internal PCI configuration registers (such as PCIC1_INRESET, PCIC1_ITADDSET, PCIC1_PSBAR) from external PCI bus agents on the PCI32 and PCI64 interfaces. The type of address map can be selected by programming bits 0-3 of “CPC0_PGCHP” on page 9-26.

The following restrictions must be observed when programming the CPC710:

- The upper 16 MB is reserved for ROM, system configuration, DMA controller, etc. See “System Register Space” on page 9-1 for the definition of this address space. Only PCI Memory spaces are allowed to overlap this area. However, they are not forwarded to the PCI bus.
- At least 1MB of system memory must be available at address 0. Minimum granularity of DIMMs is 16MB.
- System memory cannot be located above 2GB. Access in the upper 2GB is not checked by the CPC710 and result is unpredictable.
- Avoid overlapping system memory extents with PCI extents. Hang conditions and unpredictable results can occur if a processor accesses an address contained in two different extents.

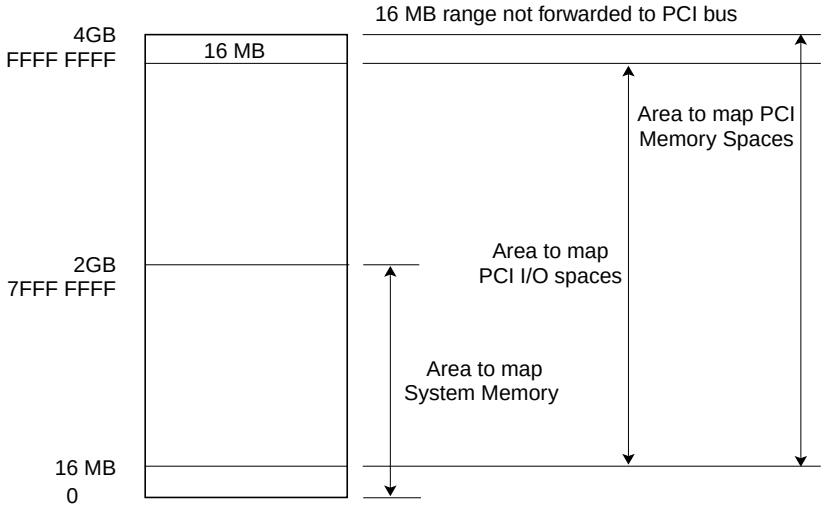


Figure 2-1. Memory Map

2.2 CPU to PCI Addressing Model

2.2.1 PREP and FPHB Modes

Programmable registers described in “Specific PCI Host Bridge Registers” on page 9-8 map PCI Memory and PCI I/O address spaces into the 4 GB System address space. Each PCI bridge in the CPC710 contains a set of these registers, allowing firmware to program PCI address spaces anywhere in memory rather than at fixed PCI address spaces.

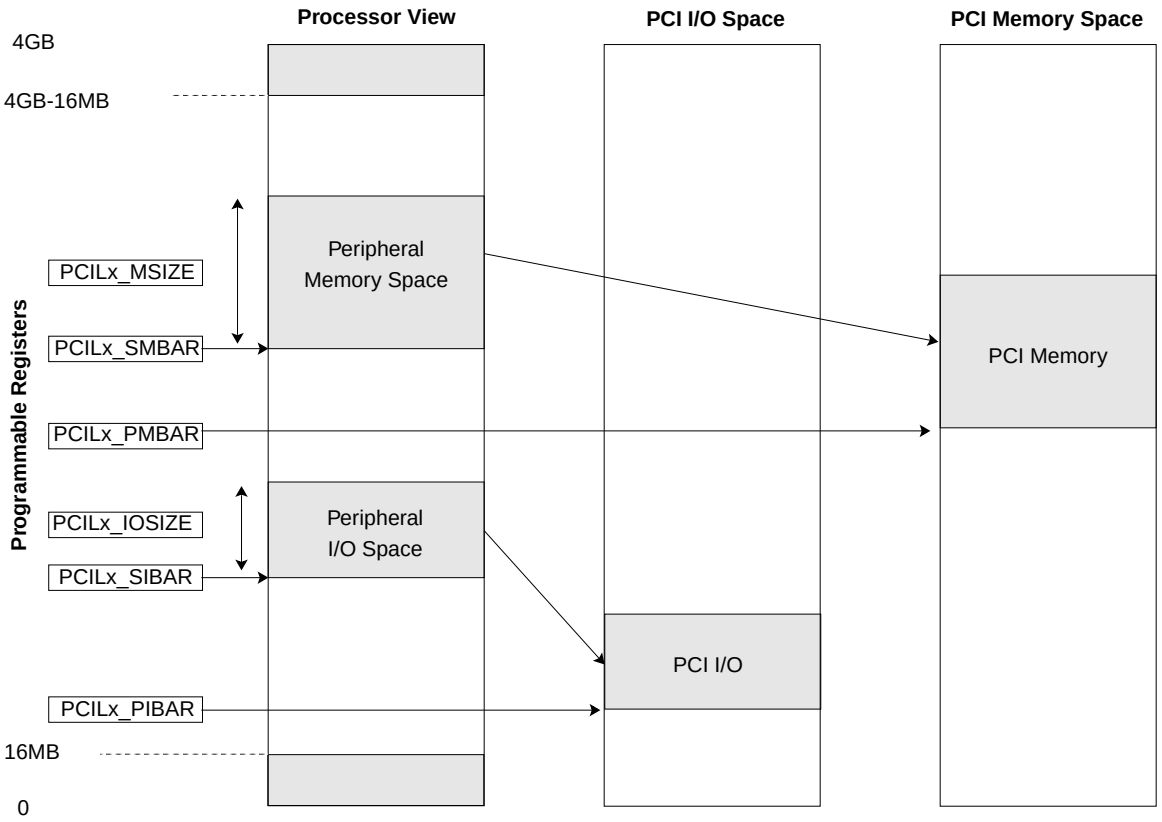


Figure 2-2. CPU to PCI Addressing Model (PREP and FPHB Modes)

As the above figure shows, the CPC710 monitors addresses on the processor bus to determine whether a CPU address falls within the ranges specified by the following PCILx_SMBAR/PCILx_MSIZE and PCILx_SIBAR/PCILx_IOSIZE registers:

- “PCILx_MSIZE” on page 9-98
- “PCILx_SMBAR” on page 9-113
- “PCILx_IOSIZE” on page 9-97
- “PCILx_SIBAR” on page 9-112

If the address falls within one of these ranges, the 60x interface logic passes the address and command to the appropriate PCI bridge logic for execution using the translation specified by the following PCILx_PMBAR or PCILx_PIBAR registers:

- “PCILx_PIBAR” on page 9-100
- “PCILx_PMBAR” on page 9-102

2.2.2 CHRP Mode

PCI Memory and PCI I/O address spaces are mapped into the 4GB System address space with the use of several programmable registers contained in “Specific PCI Host Bridge Registers” on page 9-8. These registers provide firmware a means to program PCI address spaces anywhere in memory as opposed to having fixed PCI address spaces and each PCI bridge inside the CPC710 contains a set of these registers.

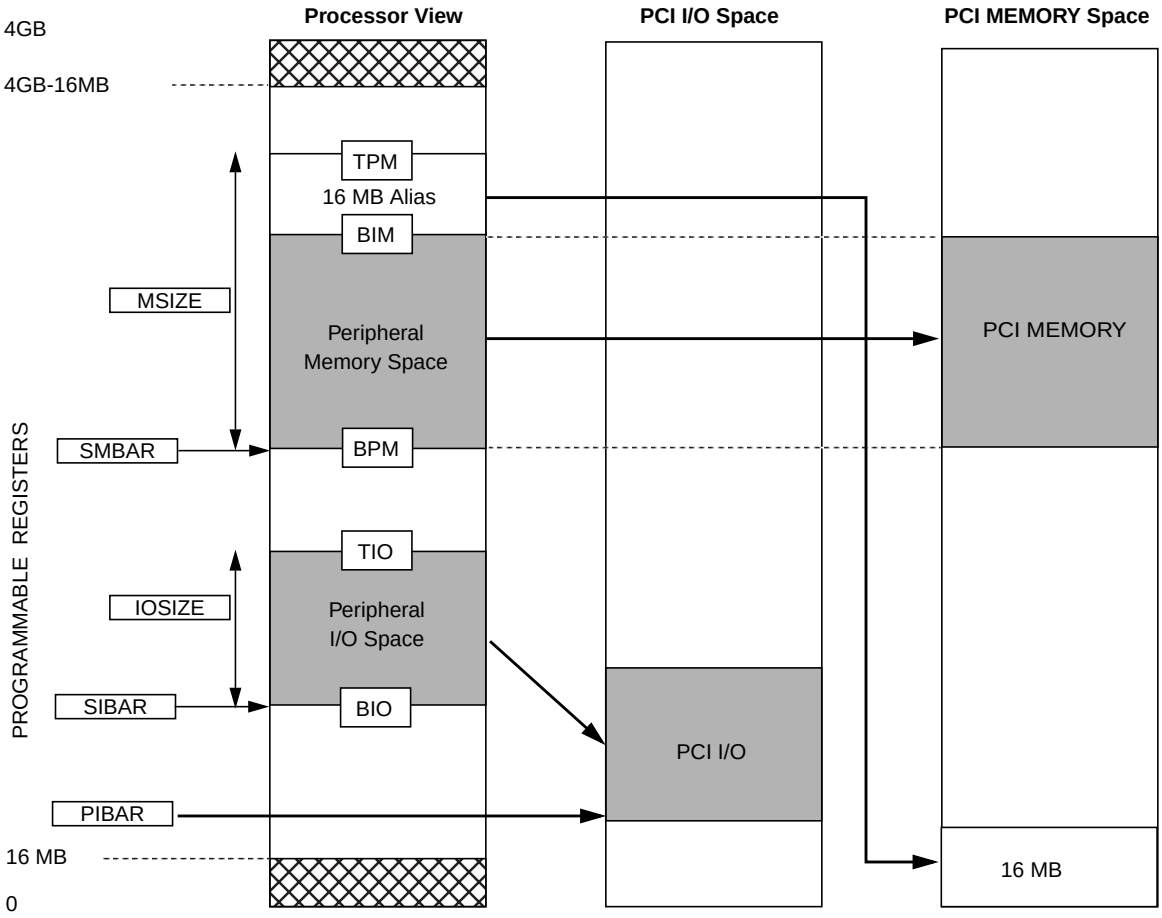


Figure 2-3. CPU to PCI Addressing Model (CHRP Mode)

As the above Figure shows, the CPC710 will monitor addresses on the processor bus and determine if a CPU address falls within the ranges specified by the PCILx_SMBAR/PCILx_MSIZE or PCILx_SIBAR/PCILx_IOSIZE registers:

- “PCILx_MSIZE” on page 9-98.
- “PCILx_SMBAR” on page 9-113

- “PCILx_IOSIZE” on page 9-97
- “PCILx_SIBAR” on page 9-112

If the address falls within one of these ranges, the 60x interface logic passes the address and command to the appropriate PCI bridge logic for it to execute with the translation specified by the PCILx_PMBAR or PCILx_PIBAR registers:

- “PCILx_PIBAR” on page 9-100
- “PCILx_PMBAR” on page 9-102

2.2.3 Peripheral I/O Address Translation

The first 8MB of Peripheral I/O space requires additional translation. To prevent 32-byte granularity accesses to ISA addresses, the CPC710 supports a noncontiguous I/O address mode in which the first 64KB of PCI bus I/O space is divided into 32byte segments spaced at 4K intervals within system memory. This mode is selected by bit 5 of “PCILx_CTRLW” on page 9-92.

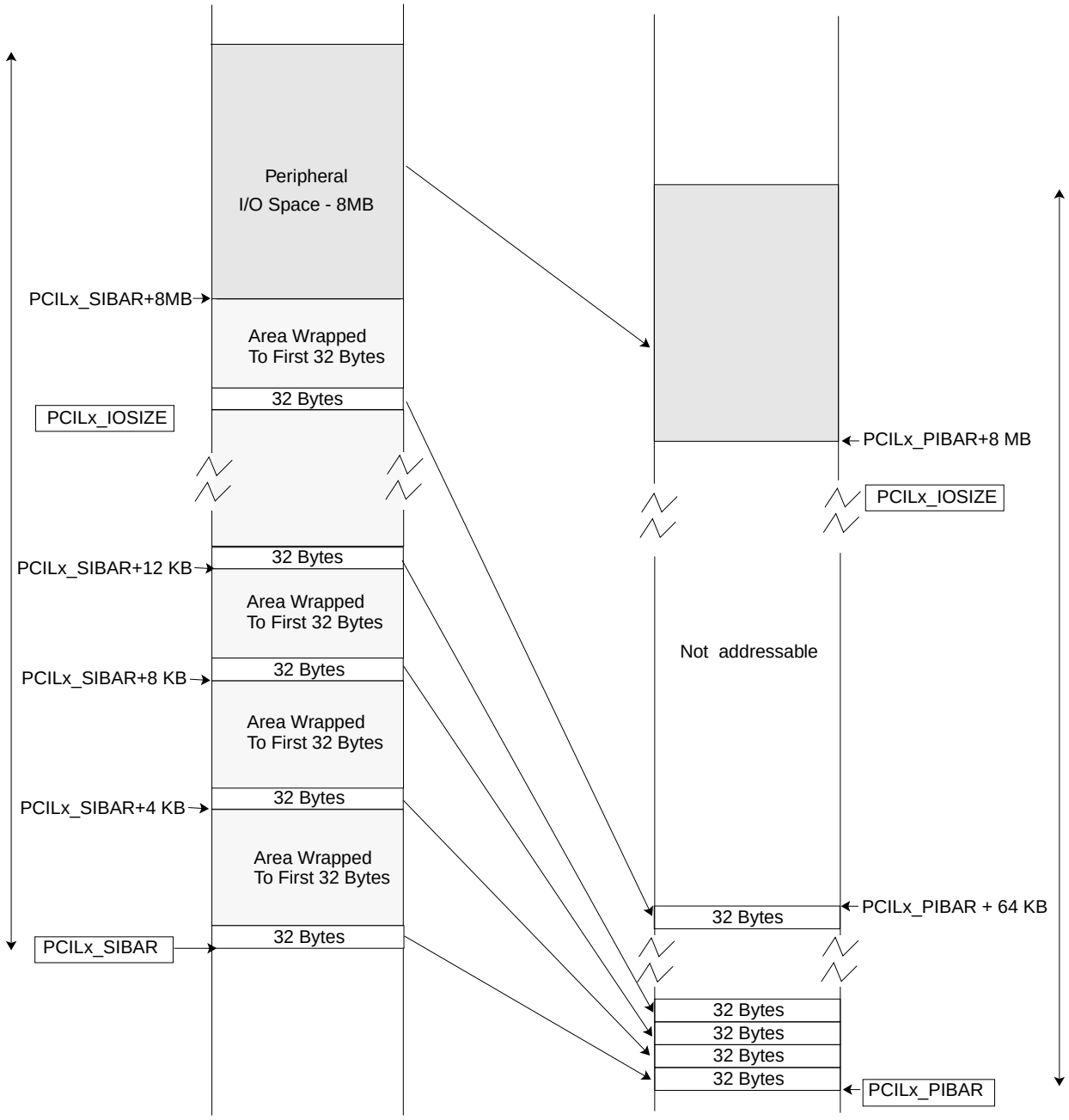


Figure 2-4. Noncontiguous I/O Address Mode Enabled

2.3 PCI to System Memory

Two types of address mapping modes are available:

- PowerPC Reference Platform (PREP) Mode
- Flexible PCI Host Bridge (FPHB) Mode.

To select a mode, program bit CPC0_PGCHP[0:1] for the PCI32 or bit CPC0_PGCHP[2:3] for the PCI64 in “CPC0_PGCHP” on page 9-26.

2.3.1 PowerPC Reference Platform (PREP) Mode

In PREP Mode, access from the PCI to the system can be performed with or without PCI address translation. When translation is used, the most common method is to translate addresses by complementing the upper 12 bits. PCI addresses ranging from x’8000 0000’ to x’FFFF FFFF’ are translated to system memory addresses x’0000 0000’ to x’7FFF FFFF’.

In this mode only PCI access to Memory are decoded by the CPC710; Configuration and I/O are not decoded.

2.3.1.1 PCI Master Address Operation

Whenever the PCI bridge logic identifies addresses coming from ISA Masters (when the P_ISA_MASTER signal is active =1), they are passed directly to system memory. Otherwise, the untranslated addresses are checked to determine whether they fall within a bridge’s PCI memory address range by comparing the PCI address to the following registers:

- “PCILx_PMBAR” on page 9-102
- “PCILx_MSIZE” on page 9-98

If there is no match and if translation is enabled by software, the PCI address is translated to a system address (bit 4 - “PCILx_PR” on page 9-104). A series of checks is performed to determine whether the access is back to the same bridge. If it is, the PCI bridge will not respond to the PCI master.

The PCI bridge logic also forwards the access to system memory. If this address does not match a memory configuration extent, the memory controller logic returns an invalid address error, thus ensuring that PCI masters do not access system facilities.

2.3.1.2 Translation Enabled in PREP Mode

If translation is enabled, the PCI bridge logic translates addresses before presenting them to system memory, as shown in the following figure. However, not all addresses are presented.

The translation is enabled by PCILx_PR[4] at address CPC0_PCIBAR + 0x000F 7F20.

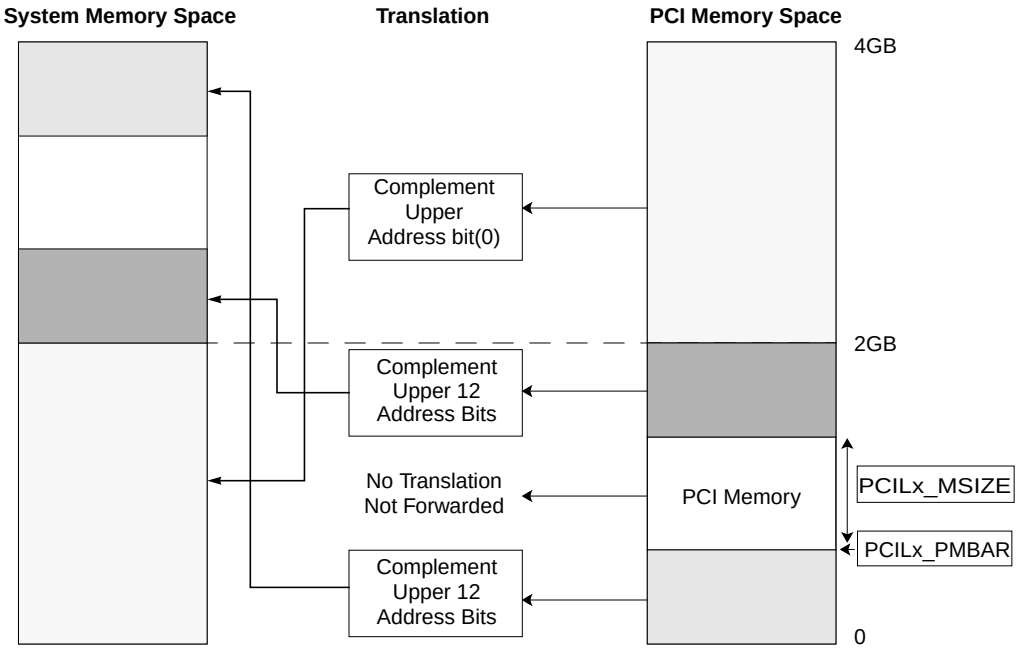


Figure 2-5. Address Translation Enabled in PREP Mode

2.3.1.3 Translation Disabled in PREP Mode

If translation is disabled, the PCI bridge does not translate addresses before presenting them to the system, as shown in the following figure.

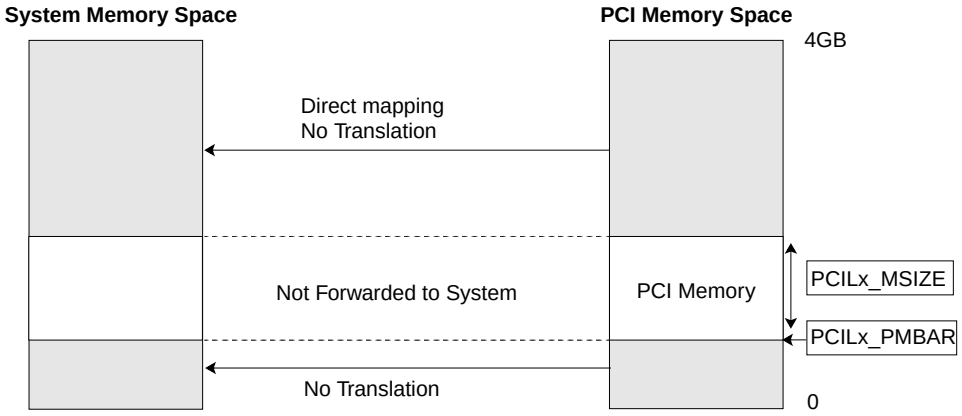


Figure 2-6. Address Translation Disabled in PREP Mode

Note: Translation can be disabled for CPU-to-PCI transfers if the values stored in the PCILx_PMBAR and PCILx_SMBAR registers are the same.

2.3.2 Flexible PCI Host Bridge (FPHB) Mode

In FPHB Mode, External Masters on the 32-bit and 64-bit PCI buses address system memory using the address translation model shown in the following figure. This model uses several programmable registers in “Specific PCI Host Bridge Registers” on page 9-8.

Note: Each PCI bridge contains a set of programmable registers.

The CPC710 monitors addresses on the PCI bus to determine whether a PCI address falls within the range specified by the following PCIC1_PSBAR/PCIL0_PSBAR/PCILx_PSSIZE registers:

- “PCIC1_PSBAR” on page 9-58
- “PCIL0_PSBAR” on page 9-80
- “PCILx_PPSIZE” on page 9-103

If an address falls within this range, the PCI interface logic passes the address to the 60x bridge logic for execution using the translation specified by “PCILx_BARPS” on page 9-85.

The memory space size can be extended up to 4 GBytes by setting bit 27 of the CPC0_PGCHP register (See page 9-27). In this case the addressing mechanism is the same as the one used for address from 0 to 2GBytes, but it uses registers PCIC1_PPBAR, PCIL0_PPBAR, and PCIL_PPSIZE for PCI address definition, and uses register PCIL_BARPP for base address in system memory.

- PCI64 Configuration by external PCI Agent:

This flexible FPHB mode allows the configuration by an external PCI agent of some of the CPC710 registers of the PCI 64 bus bridge such as the PCIC1_PSBAR, PCIC1_PPBAR, PCIC1_ITADDSET and PCIC1_INTRESET registers. See “Standard PCI Configuration Registers” on page 9-6.

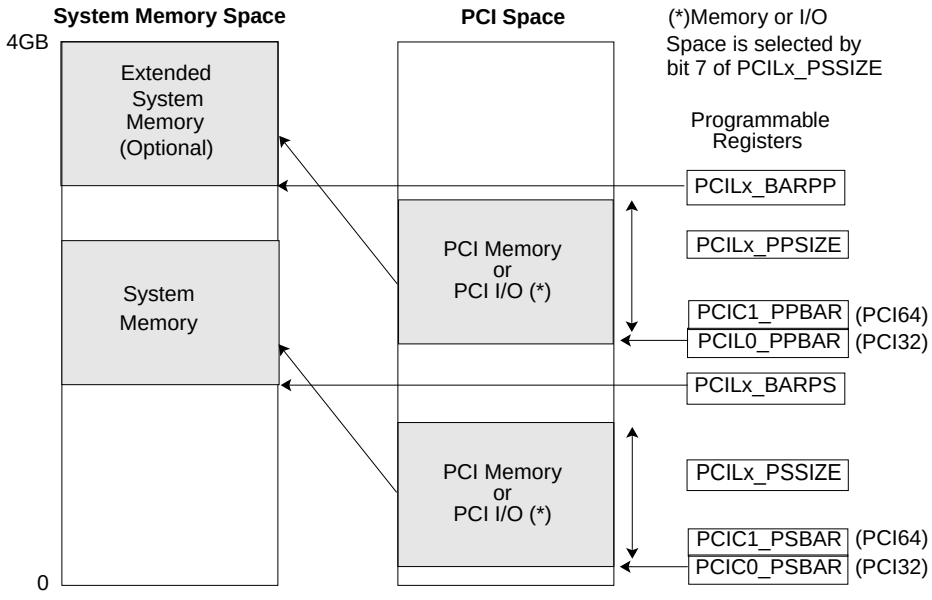


Figure 2-7. PCI to System Addressing Model (FPHB Mode)

2.3.3 CHRP Address Map

The PCI64 Host Bridge and the PPCI32 Host Bridge differs only by one feature: Only the PCI64 can access Alias and Hole Memory space.

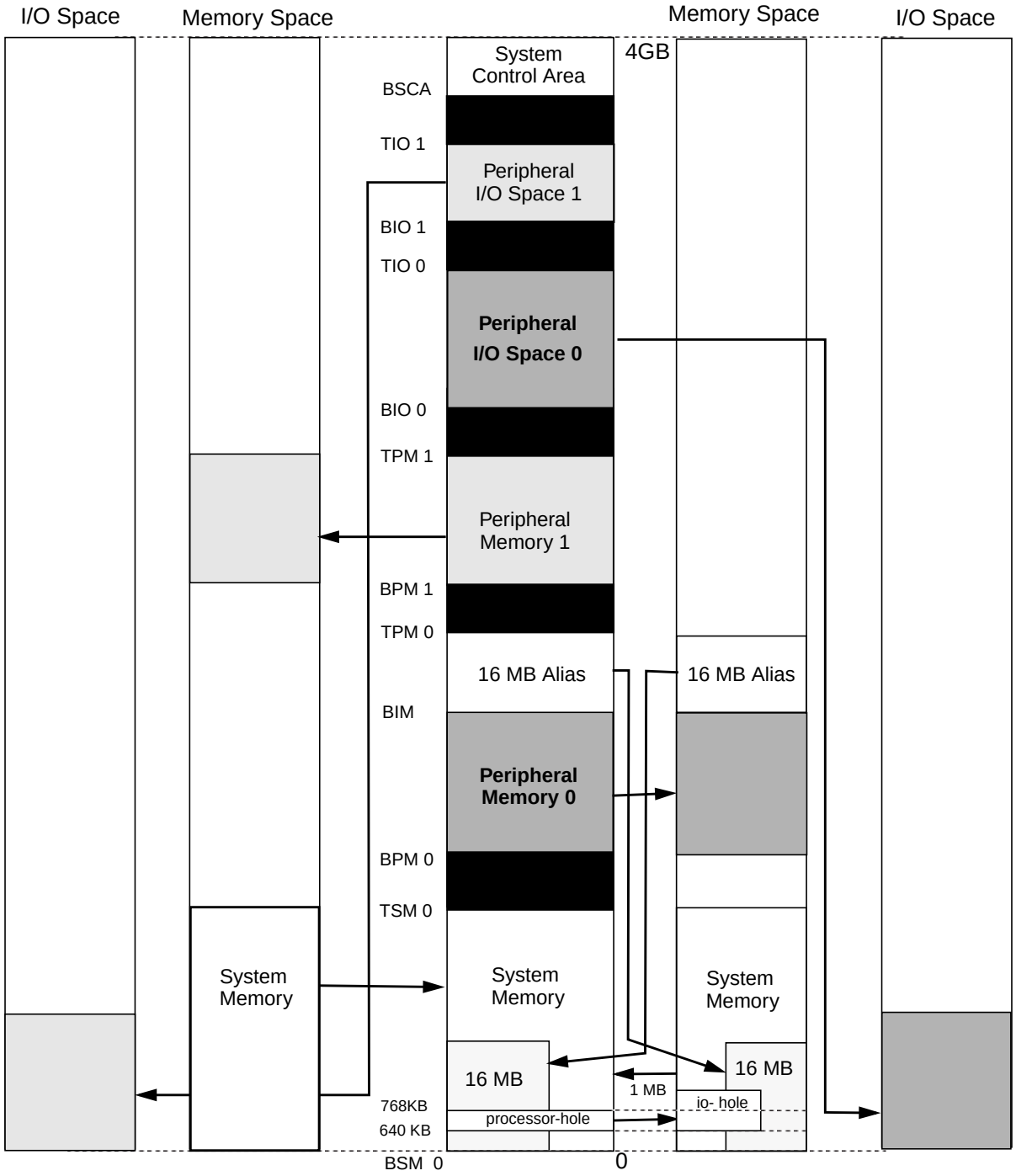


Figure 2-8. CHRP Address Map

Chapter 3. System I/O Interface

The CPC710 implements a 2 MB ROM space from address 4G-2M to 4 GB and an Extended Flash of up to 256MB.

3.1 Configuration

There is no configuration requirement for SIO logic. These areas are hard wired in the upper 16 MB of real memory.

3.2 System I/O Registers: Application Presence Detect Bits

The device provides Output Enables signals and read cycles for two external 32-bit registers. The read of the SDRAM0_SIOR0 or SDRAM0_SIOR1 results in a read of bits 0 to 31 of these register which correspond respectively to the data present on the line 31 and 0 of the PCI 32 bit A/D during the read cycle.

For descriptions of these registers, refer to:

- “SDRAM0_SIOR0” on page 9-127 controls PRES_OE0 signal
- “SDRAM0_SIOR1” on page 9-128 controls PRES_OE1 signal
- “SDRAM0_MCCR Register” on page 6-11 for the device’s supported values)

3.3 Flash Interface

3.3.1 Boot Rom

The CPC710’s Boot ROM base address is fixed at x’FFE0 0000’.

Accesses to the architected Boot ROM space within the size limit (defined in “CPC0_SIOC0” on page 9-38) are decoded as valid Boot ROM accesses. If the ROM Size parameter is larger than the actual amount of installed Boot ROM, the data will wrap. An access within the architected Boot ROM space but outside the size limit (CPC0_SIOC0 x’FF00 1020’) results in a bus timeout Machine Check error. The Boot ROM interface logic satisfies burst read requests from the processor by concatenating multiple bytes from the Boot ROM.

The CPC710 is designed to interface with 512 K, 1 Mb, 2Mb (x8) 3.3 V Flash memory with 80 to 120 ns access time. The following figure shows Boot Flash with the bits used for Address and Data on the PCI32 bus AD lines. PCI AD bits 20:0 are used for Flash Address (LSB starts at bit 0).

Bits [15:8] of the PCI32 bus AD lines are used for the 8-bit data. The Boot Flash is accessed under control of the device’s PCI32 controller to generate non-PCI cycles with FRAME not asserted. Flash is read and written by setting bit 4 (R/W) in the CPC0_UCTL Register.

The PCI access with Frame asserted has the maximum priority. The Boot Flash or Extended Flash access to the PCI 32 bus can be increased by setting biit 0 of the CPC0_SIOC1 Register.

During the Flash access the PCI bus is clocked by the System Clock.

3.3.2 Extended Boot Flash

In addition to the 2MBytes of boot Flash it is possible with the same type of operation to access to:

16, 32, 64, 128 or 256 MBytes of Extended Flash.

The Extended Flash can be Read/write by setting the bit 7 (R/W) in the CPC0_UCTL Register @ FF00 1000.

The pin : FLASH_CE_ (Extended Flash Chip Enable) controls the access of the Boot flash when =1 or the Extended Flash when =0. This signal is set to 0 after that the CPC710 decodes an address in the Extended Flash Space.

For a fine tuning of the Flash and the System bus frequency, the Timing parameters can be modified in Register: CPC0_SIOC0 @ FF00 1020.

The Extended Flash parameters are programmable in the Register: CPC0_SIOC1 @ FF00 1090

- Defines a Base address of 8 bits aligned to the size of the Flash
  Permits to map the Flash anywhere in the 4 GB processor address space
  The address on the PCI bus is defined from 0 Lsb to 28 Msb.
- Defines size: 16, 32, 64, 128 or 256 MBytes
- Defines size of the bus 8, 16, or 32 bits
  PCI AD 15: 8 for the 8 bit
  PCI AD 23:8 for the 16 bit
  PCI AD 31:0 for the 32 bit

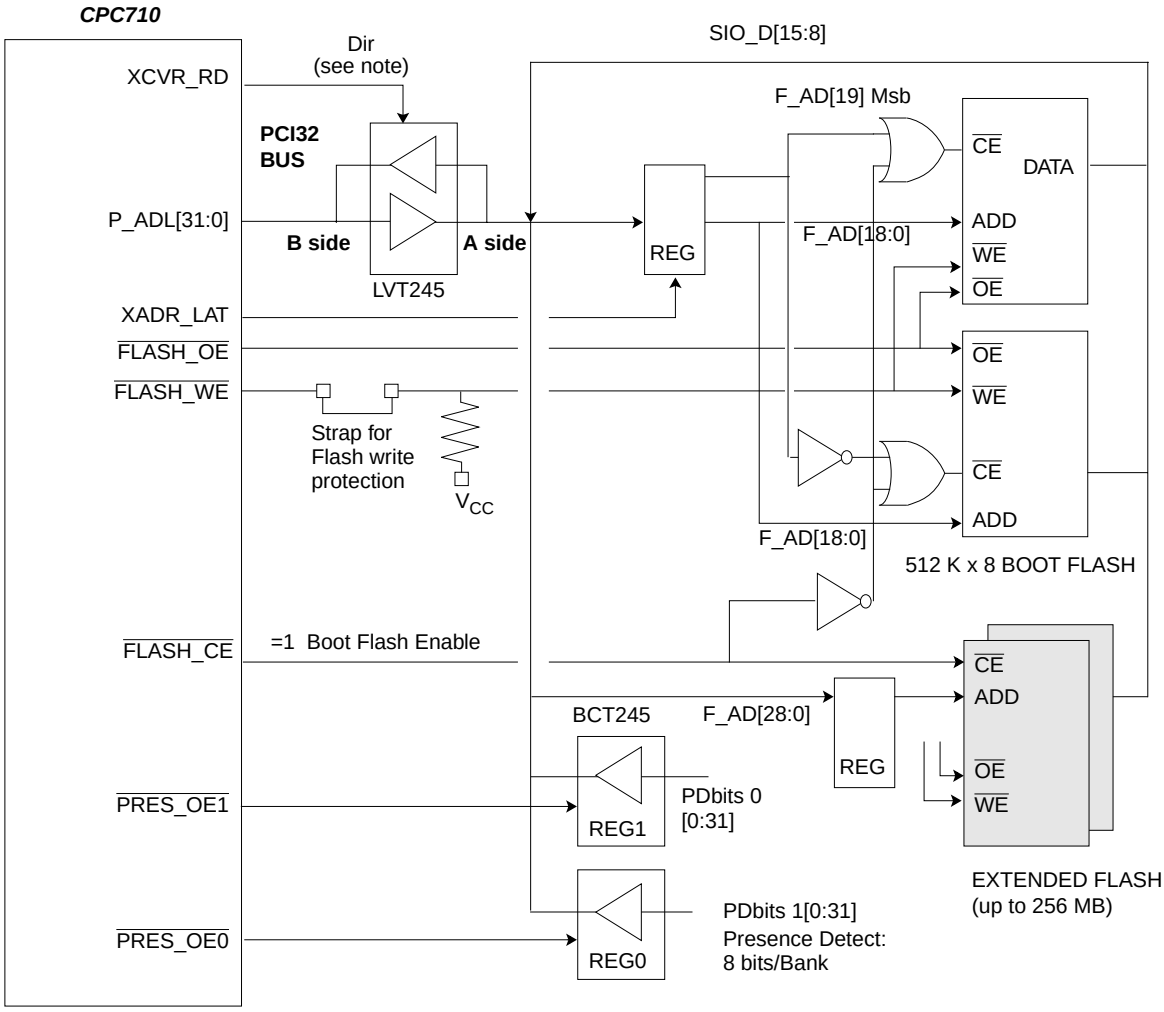
3.3.3 Byte ordering in the Boot & Extended Flash for 8 bit bus size:

The bytes in the boot Flash are ordered as following:

(Add Offset for the Flash address)
Address b'000 Byte 00 MSB
Address b'001 Byte 11
Address b'010 Byte 22
Address b'011 Byte 33
Address b'100 Byte 44
Address b'101 Byte 55
Address b'110 Byte 66
Address b'111 Byte 77 LSB

The result on the PowerPC bus is:

SYS_DATA[0:63] = 00 11 22 33 44 55 66 77



Note: There is no output enable control for the LVT245 drivers. All control is done by the Direction control bit on signal XCVR_RD
For a Read of the Boot ROM/Extended Flash data, the XCVR_RD signal is at Up level =1 such that the Data is transferred from the A Side (Flash) to the B Side (CPC710 PCI32 AD bus).

The 245 Buffer is recommended to limit the loading on the PCI 32 bus.

Figure 3-1. Connection of Boot ROM and System I/O Registers to Device

Chapter 4. 60x Interface

The 60x interface ties the CPC710 to the PowerPC 60x system bus. It performs the following functions:

- Arbitration
- Configuration
- Processor load/store address decoding
- PCI to Memory access Snoop operations
- Sync/EIEIO processing
- Endian translation
- Reset logic operations
- Time base functions

4.1 Endian Support

The Data in a system built with the CPC710 are in the following mode:

- System Memory: Big Endian
- PCI space: Little Endian (Bytes are always swapped inside the CPC710)
- PowerPC Processor Big Endian.

However, the Little Endian mode is also supported for the processor but the CPC710 internally swap bytes and unmundge address before sending it to the memory or the PCI bus.

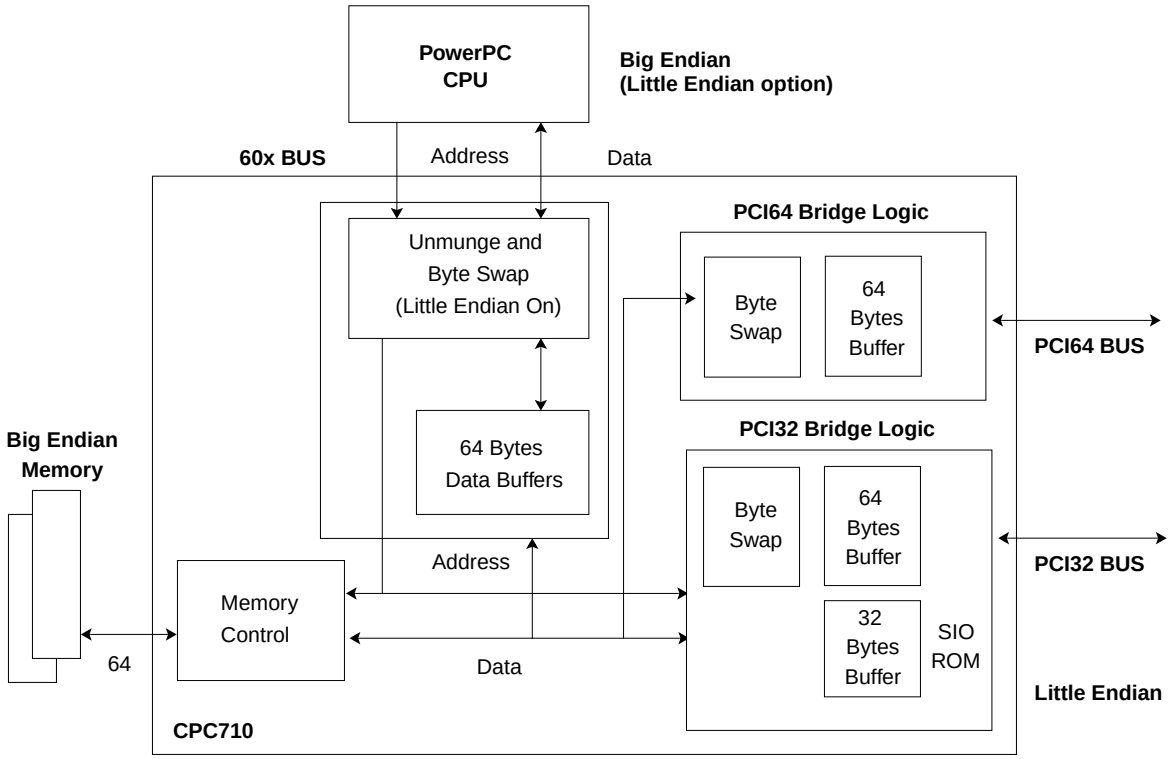


Figure 4-1. CPC710 Endian Logic

The following listing shows how the data are transmitted from the CPU to the PCI32 bus for various size of bytes.

Access CPU to PCI32 in Write

```
========================================================================
CPU Addr       data[0:63]     PCI Addr    data[31:0]    BE
========================================================================
TSIZE = 1 Byte
   0  11 00 00 00 00 00 00 00        0  00 00 00 11   1110
   1  00 11 00 00 00 00 00 00        0  00 00 11 00   1101
   2  00 00 11 00 00 00 00 00        0  00 11 00 00 1011
   3  00 00 00 11 00 00 00 00        0  11 00 00 00 0111
   4  00 00 00 00 11 00 00 00        4  00 00 00 11 1110
   5  00 00 00 00 00 11 00 00        4  00 00 11 00 1101
   6  00 00 00 00 00 00 11 00        4  00 11 00 00 1011
   7  00 00 00 00 00 00 00 11        4  11 00 00 00 0111

TSIZE = 2 Bytes
   0  11 22 00 00 00 00 00 00        0  00 00 22 11  1100
   1  00 11 22 00 00 00 00 00        0  00 22 11 00  1001
```

```
   2  00 00 11 22 00 00 00 00         0  22 11 00 00  0011
   3  00 00 00 11 22 00 00 00 BURST   0  11 00 00 00  0111
                                      4  00 00 00 22  1110
   4  00 00 00 00 11 22 00 00         4  00 00 22 11  1100
   5  00 00 00 00 00 11 22 00         4  00 22 11 00  1001
   6  00 00 00 00 00 00 11 22         4  22 11 00 00  0011

TSIZE = 3 Bytes
   0  11 22 33 00 00 00 00 00         0  00 33 22 11  1000
   1  00 11 22 33 00 00 00 00         0  33 22 11 00  0001
   2  00 00 11 22 33 00 00 00 BURST   0  22 11 00 00  0011
                                      4  00 00 00 33  1110
   3  00 00 00 11 22 33 00 00 BURST   0  11 00 00 00  0111
                                      4  00 00 33 22  1100
   4  00 00 00 00 11 22 33 00         4  00 33 22 11  1000
   5  00 00 00 00 00 11 22 33         4  33 22 11 00  0001

TSIZE = 4 Bytes
   0  11 22 33 44 00 00 00 00         0  44 33 22 11  0000
   1  00 11 22 33 44 00 00 00 BURST   0  33 22 11 00  0001
                                      4  00 00 00 44  1110
   2  00 00 11 22 33 44 00 00 BURST   0  22 11 00 00  0011
                                      4  00 00 44 33  1100
   3  00 00 00 11 22 33 44 00 BURST   0  11 00 00 00  0111
                                      4  00 44 33 22  1000
   4  00 00 00 00 11 22 33 44         4  44 33 22 11  0000

TSIZE = 5 Bytes
   0  11 22 33 44 55 00 00 00 BURST   0  44 33 22 11  0000
                                      4  00 00 00 55  1110
   1  00 11 22 33 44 55 00 00 BURST   0  33 22 11 00  0001
                                      4  00 00 55 44  1100
   2  00 00 11 22 33 44 55 00 BURST   0  22 11 00 00  0011
                                      4  00 55 44 33  1000
   3  00 00 00 11 22 33 44 55 BURST   0  11 00 00 00  0111
                                      4  55 44 33 22  1000
TSIZE = 6 Bytes
   0  11 22 33 44 55 66 00 00 BURST   0  44 33 22 11  0000
                                      4  00 00 66 55  1100
   1  00 11 22 33 44 55 66 00 BURST   0  33 22 11 00  0001
                                      4  00 66 55 44  1000
   2  00 00 11 22 33 44 55 66 BURST   0  22 11 00 00  0011
                                      4  66 55 44 33  0000
TSIZE = 7 Bytes
   0  11 22 33 44 55 66 77 00 BURST   0  44 33 22 11  0000
                                      4  00 77 66 55  1000
   1  00 11 22 33 44 55 66 77 BURST   0  33 22 11 00  0001
                                      4  77 66 55 44  0000
```

```
TSIZE = 8 Bytes
   0  11 22 33 44 55 66 77 88 BURST  0  44 33 22 11  0000
                                  4  88 77 66 55  0000
=====================================================================
```

4.1.1 PowerPC Processor Behavior Mode

The CPC710 supports PowerPC 604 and 750 processors operating in Big Endian (BE) and Little Endian (LE) modes. The mode determines the order in which a multibyte scalar is stored in memory or I/O. In BE mode, the specified address contains the scalar's most significant byte (MSB), the next sequential address contains the second MSB, and so on. In LE mode, the specified address contains the scalar's least significant byte (LSB), the next sequential address contains the second LSB, and so on.

4.1.1.1 Processor Behavior in LE Mode

PowerPC 604 and 750 processors normally operate in BE mode. To operate in LE mode, the processors generate an LE address internally and then modify, or "munge," the three low-order address bits to create a BE address equivalent. The processors do not issue unaligned LE transfers on the bus. Instead, they take an alignment interrupt. However, the PowerPC 604+ processor does issue unaligned LE transfers as long as they do not cross word boundaries. The following table describes the addresses generated by the processor for LE transfers.

4.1.1.2 Endian Behavior

PREP architecture requires data to be stored in the same Endian mode as the processor. Therefore, the CPC710 implements logic to "unmunge" the address and byte swap the data bus as it comes from the processor before sending it to memory or to the PCI bridges. See CPC710 Endian Logic shown previously.

Table 4-1. Processor Little Endian Address Modification

Processor's Internally Generated LE Effective Address [29:31]	Transfer Size (bytes)							
	1	2	3[1]	4	5[2]	6[2]	7[2]	8
	Resulting Processor Big-endian Address [29:31]							
0	7	6	5[3]	4				0
1	6	5[3]	4[3]					
2	5	4						
3	4							
4	3	2	1[3]	0				
5	2	1[3]	0[3]					
6	1	0						
7	0							

1. The PowerPC 604+ does not support 3-byte transfers in LE mode, however, these transfer sizes will result from an unaligned 4-byte access to an odd address
2. These transfer sizes are not supported by any of the processors.
3. These cells apply only to the PowerPC 604+ which performs unaligned LE transfers.

Because the CPC710 cannot determine the processor's Endian state, software must write to the Arbiter Control Register (bit 9) at the same time the processor HID register bit is updated. If the processor is operating in BE mode, bit 9 must be set to 0 to prevent the CPC710 from unmunging or byte swapping the processor's data. If the processor is operating in LE mode, bit 9 must be set to 1 to unmunge the address as specified in Processor Little Endian Address Unmunge Equations below, and to swap the data bus bytes as specified in Processor Data Bus Byte Swap for Little Endian below.

Table 4-2. Processor Little Endian Address Unmunge Equations

Transfer Size	Equation to Convert to Address
1 Byte	ADDR[29:31] XOR '111'
2 Byte	ADDR[29:31] XOR '110' and '1($\overline{31}$)1'
3 Byte	ADDR[29:31] XOR '101'
4 Byte	ADDR[29:31] XOR '100'
8 Byte	none

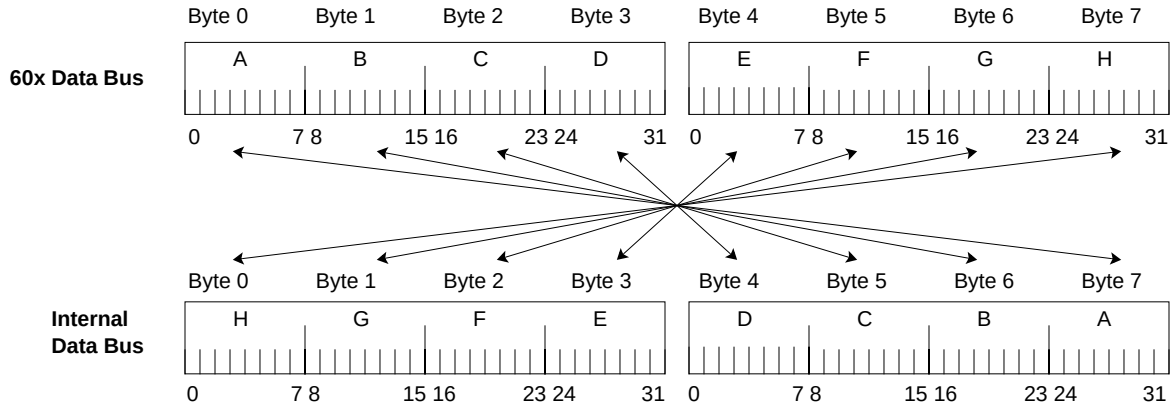


Figure 4-2. Processor Data Bus Byte Swap for Little Endian

4.2 60x Bus Arbiter Description

The arbiter in the CPC710 has the following characteristics:

- Arbitration for three devices; two levels for external masters and one for internal CPC710 requests
- No half-cycle precharge required for $\overline{\text{SYS\_TA}}$, $\overline{\text{SYS\_TEA}}$, $\overline{\text{ABB}}$, and $\overline{\text{DBB}}$
- Highly programmable address pipeline control
- Data streaming capability for external devices
- Programmable address bus parking capability
- Programmable timing on $\overline{\text{SYS\_AACK}}$
- Rotating address bus request priority scheme

4.2.1 Rotating Priority Resolution

The CPC710's 60x arbiter implements an algorithm that rotates priorities when the address bus is granted to a master. When multiple masters request the bus, the arbiter grants the bus to the master with the highest priority during the arbitration period, then downgrades that priority to the lowest level for the next period. The arbitration period occurs during the $\overline{\text{SYS\_AACK}}$ assertion cycle.

If two masters continuously request the bus, they receive alternate control. This logic is satisfactory unless a master implements a 64-byte cache line and needs to issue two 32-byte burst transfers to fill the cache. In this case, the CPC710 has a programmable mode whereby the arbiter allows one bus master to perform a pair of back-to-back address tenures even if another master requests the bus. This mode allows the CPC710's memory controller to remain in page mode for these accesses. Without this mode, another master could insert a memory transaction to take the memory controller out of page mode.

4.2.2 Address Bus Pipelining

Pipelining is controlled by bits 0 and 1 of the 60x Arbitration Control register (CPC0_ABCNTL).

Bit		Description
0-1	'10'	If enabled by software, the arbiter maintains up to a two level pipeline **per master**. The arbiter continues to grant the address bus to a specific master until there are as many as three outstanding address tenures waiting for a data bus tenure to complete or begin. Since the CPC710 supports two masters on the system bus, there can be as many as six address tenures on the 60x system bus that have not completed or begun a data bus tenure. The arbiter stops granting the address bus to a particular master after its third address tenure. The CPC710 can also drive a seventh, address-only, tenure onto the bus to satisfy a DMA snoop operation.
	'01'	The arbiter maintains a one level pipeline per master. The CPC710 stops granting the address bus to a master after it has two outstanding address tenures waiting for a data bus tenure to complete. With two masters in the system, there could be as many as four outstanding address tenures waiting for a data bus tenure to complete or begin, and a fifth CPC710 generated address-only tenure.
	'00'	Pipelining is completely disabled. Even with two masters in the system, there will only be one address tenure waiting for a data tenure to complete.
	'11'	Implemented to accommodate slave devices like an L2 lookaside that can only support one level pipeline regardless of the number of masters on the 60x bus.

4.2.3 Arbiter Requirements

4.2.3.1 Internal $\overline{\text{ABB}}$

All devices on the 60x bus must generate an internal $\overline{\text{ABB}}$. Because the arbiter may grant the address bus to a requesting device while another master is active, the requesting master must generate an $\overline{\text{ABB}}$ based on $\overline{\text{SYS\_TS}}$ and $\overline{\text{SYS\_AACK}}$. The current master does not provide an $\overline{\text{ABB}}$.

4.2.3.2 Qualified $\overline{\text{SYS\_BG}}$ Equation

Use the following equation to detect a qualified bus grant using positive logic:

$$QBG = \overline{\text{SYS\_BG}} + \overline{\text{ABB}} + \overline{\text{SYS\_ARTRY}}$$

where $\overline{\text{ABB}}$ represents the interval between $\overline{\text{SYS\_TS}}$ and $\overline{\text{SYS\_AACK}}$ active.

Bus Request ($\overline{\text{SYS\_BR}}$) need not be active to detect a qualified bus grant (parked case).

4.2.3.3 SYS_TS Assertion

All master devices must drive SYS_TS active in the cycle immediately following a qualified address bus. Otherwise, the address tenure is aborted and another master is free to drive the address bus.

4.2.3.4 SYS_BR Negation

All master devices must negate SYS_BR for at least one bus cycle immediately after receiving a qualified bus grant.

4.2.3.5 Qualified SYS_DBG Equation

The equation for qualified SYS_DBG using positive logic is:

QDBG = SYS_DBG + ARTRY

DBB is unused because the arbiter does not issue a SYS_DBG when DBB is active. The arbiter monitors transaction sizes to determine the end of a data bus tenure and waits until the previous data tenure is complete before issuing a SYS_DBG to the next master.

Note: QDGB can only be negated by an SYS_ARTRY of the address tenure associated with the QDBG data bus tenure. Therefore, once the SYS_ARTRY window has passed for an address tenure, the data bus tenure associated with that address tenure cannot be negated by SYS_ARTRY from a subsequent address tenure.

4.2.3.6 High Impedance After SYS_TEA

Masters and slaves must execute all data bus signals as high impedance within two bus clocks from SYS_TEA assertion.

4.2.3.7 SYS_DRTRY Assertion

Slaves are not allowed to drive SYS_DRTRY active. The CPC710 arbiter does not receive SYS_DRTRY.

4.2.3.8 Slave Data Bus Determination

To determine whether the data bus is currently in use by a previous address tenure, a slave must sample DBB from its master during the TS active cycle. If DBB is active, the slave must wait for DBB to go inactive in a one-level pipeline mode for at least one cycle before providing read data or accepting write data.

4.2.3.9 SYS_L2_Hit Assertion

For the CPC710 to determine whether an addressing error has occurred, all slaves on the 60x bus must assert SYS_L2_HIT when selected by an address on the 60x bus.

Warning: The SYS_L2_HIT signal is subject to timing constraints.

4.2.4 Bus Enhancements

4.2.4.1 $\overline{\text{DBB}}$ not Required by Masters

Masters do not require $\overline{\text{DBB}}$. The arbiter does not grant the data bus to a requesting master if the bus is currently in use. The CPC710 does not drive $\overline{\text{DBB}}$ since it acts as an address-only bus master.

4.2.4.2 Half-Cycle Precharge not Required on $\overline{\text{SYS\_TA}}$

The CPC710 can be programmed so the precharge of $\overline{\text{SYS\_TA}}$ is not required. This requires all slaves to initially drive $\overline{\text{SYS\_TA}}$ active or inactive immediately following a data bus grant. At the end of a data bus tenure, a slave does not perform a precharge, which requires a slave in the next data bus tenure to drive $\overline{\text{SYS\_TA}}$ in the first cycle of the tenure.

The CPC0_ABCNTL[13] mode bit, as described in “CPC0_ABCNTL” on page 9-10, forces the CPC710 to do a precharge if a slave on the 60x bus does not support this function.

Half-Cycle Precharge not Required on $\overline{\text{SYS\_TEA}}$

The arbiter does not grant the data bus for two bus cycles following assertion of $\overline{\text{SYS_TEA}}$s. This allows a slave to perform a full cycle precharge on $\overline{\text{SYS_TEA}}$s.

4.2.4.3 $\overline{\text{SYS\_ARTRY\_PREV}}$ in QDBG Equation Eliminated

When $\overline{\text{SYS\_ARTRY}}$ is asserted, the arbiter negates all bus grants in the cycle following $\overline{\text{SYS\_ARTRY}}$. This supplants the requirement for masters to qualify associated bus grants by asserting $\overline{\text{SYS\_ARTRY}}$ in a previous cycle.

4.2.5 60x Bus Transfer Types and Sizes

The following tables describe the transaction types supported by the CPC710 on the processor bus.

The CPC710 supports the PowerPC 604 critical double word burst transactions.

Table 4-3. Non-Burst Transactions ($\overline{\text{SYS\_TBST}}$ = 1)

SYS_TSIZ[0:2]	A[29:31]	Definition	Note
0 0 0	000 - 111	8-byte transfer	1
0 0 1	000 - 111	1-byte transfer	
0 1 0	000 - 111	2-byte transfer	
0 1 1	000 - 111	3-byte transfer	
1 0 0	000 - 111	4-byte transfer	
1 0 1	000 - 111	5-byte transfer	
1 1 0	000 - 111	6-byte transfer	
1 1 1	000 - 111	7-byte transfer	

1. For transfers where the number of bytes to transfer cross a doubleword boundary, the CPC710 will truncate the transfer size to avoid crossing a doubleword boundary.

Table 4-4. Burst Transactions ($\overline{\text{SYS\_TBST}}$ = 0)

SYS_TSIZ[0:2]	A[27:31]	Definition	Support	Notes
0 0 0	x x x x x	8 Byte transfer	No	2
0 0 1	0 0 x x x	16-byte transfer beginning on 32-byte boundary	TBD	
	0 1 x x x	16-byte transfer beginning on odd doubleword boundary	No	
	1 0 x x x	16-byte transfer beginning on odd 16-byte boundary	TBD	
	1 1 x x x	16-byte transfer beginning on odd doubleword boundary	No	
0 1 0	0 0 x x x	32-byte transfer beginning on 32-byte boundary	Yes	
	0 1 x x x	32-byte transfer beginning on odd doubleword boundary		1,3
	1 0 x x x	32-byte transfer beginning on odd 16-byte boundary		1,3
	1 1 x x x	32-byte transfer beginning on odd doubleword boundary		1,3

1. For transfers that cross a 32-byte boundary, the CPC710 will wrap to the beginning of the 32-byte block to satisfy the data transfer.
2. Unpredictable results will occur if this transfer size is attempted on the processor bus.
3. Not supported on store operations

Table 4-5. Transfer Types (Page 1 of 2)

TT[0:4]	Operation	Transaction	Support as Master	Support as Slave
0 0 0 0 0	Clean Sector	Address-only	Yes	NOP
0 0 0 0 1	LARX Reservation Set	Address-only	No	NOP
0 0 0 1 0	Write with Flush	SBW or Burst	No	Yes
0 0 0 1 1	Reserved (arbiter will assume address-only transaction)			
0 0 1 0 0	Flush Sector	Address-only	Yes	NOP
0 0 1 0 1	Reserved (arbiter will assume address-only transaction)			
0 0 1 1 0	Write with Kill	Burst	No	Yes (treated as 00010)
0 0 1 1 1	Reserved (arbiter will assume address-only transaction)			
0 1 0 0 0	SYNC	Address-only	No	Yes
0 1 0 0 1	TLBSYNC	Address-only	No	Yes
0 1 0 1 0	Read	SBR or Burst	No	Yes
0 1 0 1 1	RWNITC - Read with no Intent to Cache	SBR or Burst	Yes (See CPC0_ATAS Register)	Yes (treated as 01010)
0 1 1 0 0	Kill Sector	Address-only	Yes	NOP
0 1 1 0 1	ICBI	Address-only	No	NOP
0 1 1 1 0	RWITM - Read with Intent to Modify	Burst	Yes (See CPC0_ATAS Register)	Yes (treated as 01010)
0 1 1 1 1	Reserved (arbiter will assume address-only transaction)			
1 0 0 0 0	EIEIO	Address-only	No	Yes
1 0 0 0 1	Reserved (arbiter will assume address-only transaction)			

Note: SBW = Single Beat Write, SBR = Single Beat Read

Table 4-5. Transfer Types (Page 2 of 2)

TT[0:4]	Operation	Transaction	Support as Master	Support as Slave
1 0 0 1 0	Write with Flush Atomic	SBW	No	Yes (treated as 00010)
1 0 0 1 1	Reserved (arbiter will assume address-only transaction)			
1 0 1 0 0	ECOWX - Graphics Write	SBW	No	Yes
1 0 1 0 1	Reserved (arbiter will assume address-only transaction)			
1 0 1 1 x				
1 1 0 0 0	TLB Invalidate	Address-only	No	
1 1 0 0 1	Reserved (arbiter will assume address-only transaction)			
1 1 0 1 0	Read Atomic	SBR or Burst	No	Yes (treated as 01010)
1 1 0 1 1	Reserved (arbiter will assume address-only transaction)			
1 1 1 0 0	ECIWX - Graphics Read	SBR	No	Yes
1 1 1 0 1	Reserved (arbiter will assume address-only transaction)			
1 1 1 1 0	RWITM Atomic	Burst	No	Yes (treated as 01010)
1 1 1 1 1	Reserved (arbiter will assume address-only transaction)			

Note: SBW = Single Beat Write, SBR = Single Beat Read

4.3 Data Gathering

The 60x logic gathers data for CPU store-transfers to the PCI bus bridges. During data gathering, single beat stores of up to 32bytes from the CPU are gathered before being sent to the PCI bus bridge unit. Data gathering reduces asynchronous boundary crossings and facilitates data bursting on the PCI bus.

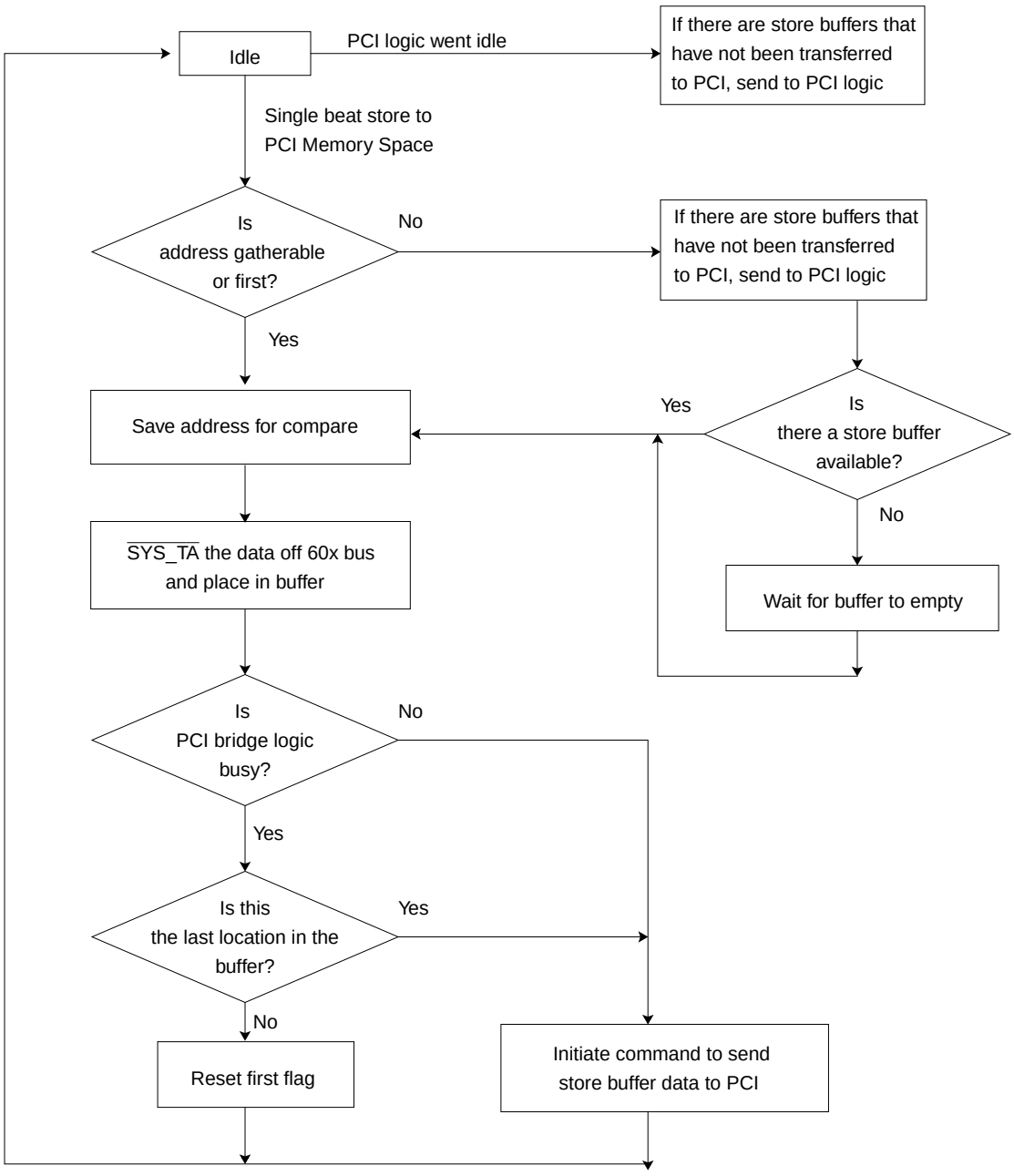

Idle
PCI logic went idle
If there are store buffers that have not been transferred to PCI, send to PCI logic
Single beat store to PCI Memory Space
Is address gatherable or first?
No
If there are store buffers that have not been transferred to PCI, send to PCI logic
Yes
Save address for compare
Yes
Is there a store buffer available?
No
SYS_TA the data off 60x bus and place in buffer
Wait for buffer to empty
Is PCI bridge logic busy?
No
Yes
Is this the last location in the buffer?
Yes
No
Reset first flag
Initiate command to send store buffer data to PCI

Figure 4-3. Data Gathering Algorithm

4.4 SYNC and EIEIO

When a processor executes a SYNC instruction, a SYNC address-only tenure is broadcast on the 60x bus to notify the system that a software-placed barrier is present. The system is responsible for ensuring all previously executed load and store operations are complete and all resultant actions are

visible to the system. The CPC710 satisfies this requirement by SYS_ARTRYing the SYNC operation until all of its store buffers are empty, all reads have been executed, and all data have been placed in internal CPC710 buffers for requests issued by the same processor issuing SYNC.

When a processor broadcasts an EIEIO on the 60x bus, the system is responsible for ensuring all previous transactions are complete before executing operations. The CPC710 does not SYS_ARTRY the EIEIO because the 60x logic dispatches bus transactions to the logic units in the order in which they occur on the system bus and each logic unit executes its commands in the order received. For diagnostic purposes, the CPC710 can be programmed to SYS_ARTRY the EIEIO in the same manner as SYNC (see bit 10 of “CPC0_ABCNTL” on page 9-10).

The logic units are system memory, PCI-32 bus bridge, PCI64 bus bridge, system I/O logic, and DMA controller logic. EIEIO operations are valid for transfers to and from the same logic unit, but execution order of load and store operations to different logic units cannot be guaranteed. For example, a store to the PCI-32 bus bridge followed by a PCI64 store could be presented to the respective PCI buses in reverse order if a bus is busy. To preserve the order among logic units, software must issue a SYNC instead of an EIEIO.

4.5 Address Retry (SYS_ARTRY)

4.5.1 Precharging SYS_ARTRY and SYS_SHD

This function is programmable by the bit 12 of CPC0_ABCNTL register.

The CPC710 can be programmed to precharge the ARTRY_ and SHD_ signals. This requires that all devices on the 60x bus must disable their precharge of this signal. The CPC710 negates these signals in the second cycle following the last cycle of the ARTRY window for 1/2 of a bus cycle

4.5.2 SYS_ARTRY Assertions

The CPC710 asserts SYS_ARTRY for:

- SYNC operations as described in the previous section
- EIEIO operations as described in the previous section
- XFERDATA when more than two transfers have been initiated
- a processor access to the PCI bus when a PCI-ISA bus bridge requests the same PCI bus
- a processor access to system memory when a DMA occurs to the same cache line
- a processor access to system memory when a DMA operation occurs to the same line
- a processor access into a range of PCI-32 or PCI64 addresses defined as potential deadlock

4.5.3 Recommended SYS_ARTRY Procedure

- A master that has had its address tenure retried should negate its SYS_BR[n] for at least one bus cycle in the cycle immediately following detection of an active SYS_ARTRY.
- A master that has retried an address tenure due to a snoop hit should activate its SYS_BR[n] in the cycle immediately following the detection of an active SYS_ARTRY. This ensures the master that retries is serviced before the master that was retried.

4.6 Deadlock Avoidance

Potential Deadlock

Potential deadlock situations can occur when the CPU attempts to access (read or write) a PCI master, while at the same time the PCI master attempts to access a cacheable main memory address. The PCI to memory access generates a snoop cycle on the processor bus, and if that results in a cache hit, the PCI master's memory access will get retried. If the PCI master (typically a PCI-PCI, PCI-ISA or PCI-VME bridge) does not have enough internal buffering to support incoming as well as outgoing transactions or uses prioritization that requires the outgoing transaction to complete before servicing the incoming CPU's read or write, a deadlock will occur.

Deadlock Avoidance with the CPC710

There are three ways to implement deadlock avoidance:

1. For the PCI32 bus only, external logic can be used with P_MEMREQ and P_MEMACK signals (not recommended for new designs).
2. External logic can be used with the $\overline{\text{DLK}}$ and $\overline{\text{NODLK}}$ signals (not recommended for new designs).
3. Use of CPC710 internal deadlock avoidance logic. This is the preferred solution, as it can be managed by software. The $\overline{\text{DLK}}$ and $\overline{\text{NODLK}}$ pins may not be supported in future revisions, and should not be used.

The internal deadlock avoidance circuit is active when at least one of the bits 0, 1, and 16 of the CPC0_ DLKCTRL register are set to 1 and CPC0_PGCHP register bit 24 is set to 0.

For each PCI bus, there are six unique registers and one common register that are associated with the deadlock avoidance logic.

- Internal Deadlock avoidance circuit control setup:

PCILx_DLKCTRL PCILx_DLKDEV

- Internal Deadlock avoidance address window definition:

PCILx_TPMDLK PCILx_BPMDLK

PCILx_TIODLK PCILx_BIODLK

- External Deadlock signal control (common for PCI32 and PCI64)

CPC0_PGCHP[24] should be set to 0 when using the internal deadlock logic.

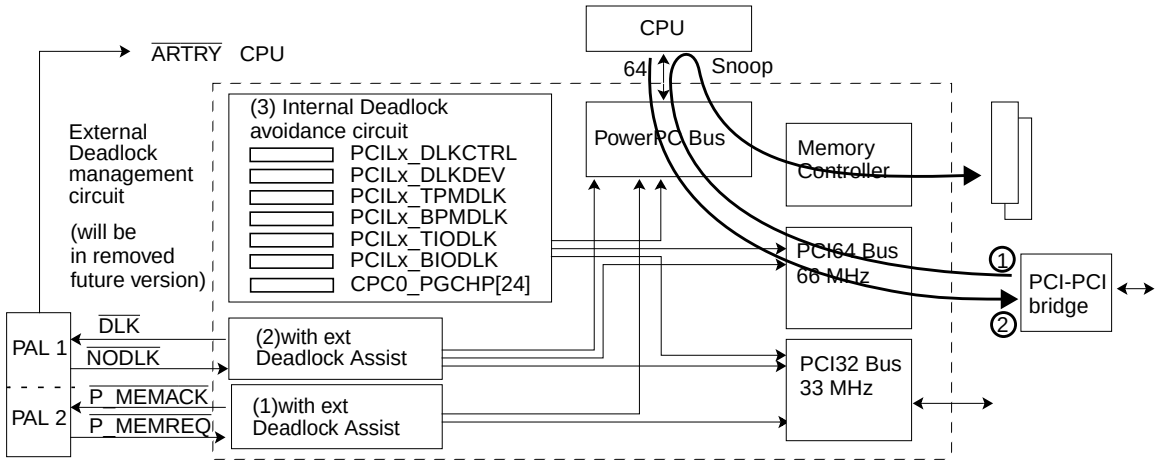


Figure 4-4. Deadlock avoidance circuits in the CPC710

4.6.1 Deadlock Avoidance Operation with Internal Logic

The Deadlock avoidance internal circuit is enabled when any one of the bits 0, 1, or 16 of the Deadlock avoidance control register (PCILx_DLKCTRL) is set to 1. An access from the CPU to PCI that hits in the range of addresses defined as the deadlock avoidance window (as programmed in registers PCILx_TPMDLK, PCILx_BPMDLK, PCILx_TIODLK, PCILx_BIODLK) will generate a $\overline{\text{SYS\_ARTRY}}$ signal on the CPU bus.

For read operations that fall in the deadlock avoidance window, the CPC710 takes the CPU request and then generates the $\overline{\text{SYS\_ARTRY}}$ and performs a delayed read.

For write operations, the first write in the address window is not retried if the PCI bus is busy. If the deadlock avoidance circuit is not programmed, the $\overline{\text{SYS\_ARTRY}}$ will not appear. This is what will happen if the PCI bus agent is smart enough to buffer its transactions in the case that resources are not free, and allowing for a PCI retry operation to occur. If system PCI agents support this type of operation then use of the CPC710's internal deadlock avoidance is not necessary.

For monitoring purposes, the $\overline{\text{DLK}}$ signal can be enabled (PCILx_DLKCTRL[6] = 1) and observed.

Sequence of operation with the Deadlock avoidance internal circuit

1. CPC710 detects that the CPU to PCI access is in the range of addresses defined in registers.
2. CPU to PCI access is flushed
3. $\overline{\text{SYS\_ARTRY}}$ is generated on the 60X bus by the CPC710
4. Execute PCI master Read/Write access to System Memory with Snoop.
5. End of the PCI master operation
6. CPU access to the PCI

4.6.2 Deadlock Avoidance Operation with External Logic

This mode of operation requires the use of external logic to generate the CPC710 input $\overline{\text{NODLK}}$. The CPC710 will generate the output $\overline{\text{DLK}}$ for a CPU access to the PCI32 or PCI64 bus when:

- the input $\overline{\text{NODLK}}$ is not active (is read as a '1')

- the access is within the address window defined by the appropriate address registers (PCILx_TPMDLK, PCILx_BPMDLK, PCILx_TIODLK, PCILx_BIODLK)

- CPC0_PGCHP[24] is set to '1'

The CPC710 will then generate a $\overline{\text{SYS\_ARTRY}}$ to the CPU bus logic when the externally generated $\overline{\text{NODLK}}$ signal becomes active (is read as a '0').

4.7 Error Handling for CPU-Initiated Transactions

The CPC710 uses Machine Checks to indicate errors. This allows software to log errors before the system is shut down. In an MP environment, the CPC710 activates the Machine Check pin that corresponds to the CPU initiating the transaction.

4.7.1 Checkstop Errors

The CPC710 generates a checkstop when the following are detected:

- Address parity error on the 60x system bus (if enabled)
- Data parity error on 60x system bus (if enabled)
- Internal timeout due to no response from slave on load

The 60x logic performs the following when generating a checkstop:

1. Sets appropriate bit(s) in CPC0_SESR

2. Drives $\overline{\text{CHKSTOP}}$ active until power on reset

The following table describes the error handling performed for CPU initiated transactions. The 60x logic drives $\overline{\text{SYS\_MCP0:1}}$ signals, not the PCI bridge logic or the memory control logic.

Table 4-6. CPU Initiated Transactions (Page 1 of 5)

Operation	Error	Mode	Action	Note
Access not directed to the CPC710	Addressing Error ($\overline{\text{SYS\_L2\_HIT}}$ not driven active)	Disabled	No action taken	
		Enabled	Set No Select error bit in CPC0_SESR Set error address in CPC0_SEAR CPC0_PGCHP[26] = 0: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ CPC0_PGCHP[26] ≠ 0: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
Notes: 1. A dummy 0 is returned for read operation. For write, data is ignored.				

Table 4-6. CPU Initiated Transactions (Page 2 of 5)

Operation	Error	Mode	Action	Note
Access to CPC710	Bus Time-out: Time expired from SYS_AACK active to first SYS_TA	Disabled	Inhibit timer; no action taken	
		Enabled	Set bus time-out error bit in CPC0_SESR Set checkstop generated bit in CPC0_SESR Set error address in CPC0_SEAR Signal Checkstop	
Access to internal CPC710 facilities	Access to a reserved or non- implemented address		Terminate CPU transaction normally	1
	Alignment or size			
	Store to read-only register			
	Load from write-only register.			
Access to system memory	Single bit error	Don't care	Set single-bit error and syndrome in SDRAM0_MESR Set error address in SDRAM0_MEAR Return corrected data to CPU If CPC0_PGCHP[17] = 1 and CPC0_PGCHP[26] = 1: Set memory error bit in CPC0_SESR Set memory error address in CPC0_SEAR Signal Machine Check with SYS_MCP0:1	
	Double-bit error	Normal	Set error in SDRAM0_MESR Set error address in SDRAM0_MEAR Set memory error bit in CPC0_SESR Set memory error address in CPC0_SEAR if CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with SYS_TEA Stores: Signal Machine Check with SYS_MCP0:1 If CPC0_PGCHP[26] = 1, signal Machine Check with SYS_MCP0:1	
		Diagnostic	Set double-bit error in SDRAM0_MESR Set error address in SDRAM0_MEAR Return uncorrected data to CPU Signal Machine Check with SYS_MCP0:1 if write less than 8 bytes	

Notes: 1. A dummy 0 is returned for read operation. For write, data is ignored.

Table 4-6. CPU Initiated Transactions (Page 3 of 5)

Operation	Error	Mode	Action	Note
Access to PCI bus	No DEVSEL received		Master-abort the PCI transaction Set master-aborted bit 13 in PCI Status register Set Devsel error bit 3 in PCILx_PLSSR register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1
	Detected SERR active during PCI transaction		Master-abort the PCI transaction Set master aborted bit 13 in PCI Status register Set SERR detected error in PCILx_PLSSR register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1

Notes: 1. A dummy 0 is returned for read operation. For write, data is ignored.

Table 4-6. CPU Initiated Transactions (Page 4 of 5)

Operation	Error	Mode	Action	Note
Access to PCI bus (cont'd)	Detected PCI bus data parity error on load	Enabled by bit 6 in PCI PCICx_CMD register	Continue transfer on PCI bus to completion Activate the PERR signal Set data parity error bit 8 in PCI Status register Set data parity error bit 15 in PCI Status register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1
		Disabled by bit 6 in PCI PCICx_CMD register.	Set parity error bit 15 in PCI Status register Proceed normally with PCI transaction Proceed normally with CPU transaction	
	Detected PERR on store	Enabled by bit 6 in PCI PCICx_CMD register	Continue transfer on PCI bus to completion Set data parity error bit 8 in PCI Status register Set data parity error bit 15 in PCI Status register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1
		Disabled by bit 6 in PCI PCICx_CMD register	Set data parity error bit 15 in PCI Status register Proceed normally with PCI transaction Proceed normally with CPU transaction	
	Detected target abort		Set received target abort bit in PCI Status register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1

Notes: 1. A dummy 0 is returned for read operation. For write, data is ignored.

Table 4-6. CPU Initiated Transactions (Page 5 of 5)

Operation	Error	Mode	Action	Note
Access to PCI bus (cont'd)	PCI Bus timeout: P/G_TRDY count expired	Enabled	Master-abort the PCI transaction Set master aborted bit 13 PCI Status register Set PCI bus time-out error in PCILx_PLSSR register Set PCI error bit in CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1
	Retry count expired	Enabled	Stop retrying PCI transfer Set retry count expired bit in PCILx_PLSSR register Set PCI error bit is CPC0_SESR Set error address in CPC0_SEAR register If CPC0_PGCHP[26] = 0: Loads: Signal Machine Check with $\overline{\text{SYS\_TEA}}$ Stores: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ If CPC0_PGCHP[26] = 1: Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Terminate CPU transaction normally	1

Notes: 1. A dummy 0 is returned for read operation. For write, data is ignored.

Chapter 5. Initialization

5.1 CPC710 Power Up Sequence:

- time= - 100 mS
  - All PLL inputs are stable and at their final values: (See following recommended setting for PLL _TUNE and PLL_RANGE input signals)
  - SYS_CLK is stable at or below the target frequency
  - VDDA and Vdd (supply) are at their final values
  - POWERGOOD input is de-asserted Low for Reset.
  - PLL_RESET input is asserted active (Low).
  - PCI clocks inputs (PCI_CLK and PCG_CLK) are stable at the target frequency
- time= 0
  - PLL_RESET input is de-asserted inactive (High).
  - PLL_LOCK output goes down up to the time that clock are locked to the PLL, then is asserted active (High), indicating the PLL is locked.
- time= 500 µs minimum
  - POWERGOOD input is asserted (High).
  - HRESET output de-asserted by the CPC710 (High)
  - Bus transactions may begin.
  - Boot can begin.

Note 1: Chip reset is only controlled by the SYS_CLK.

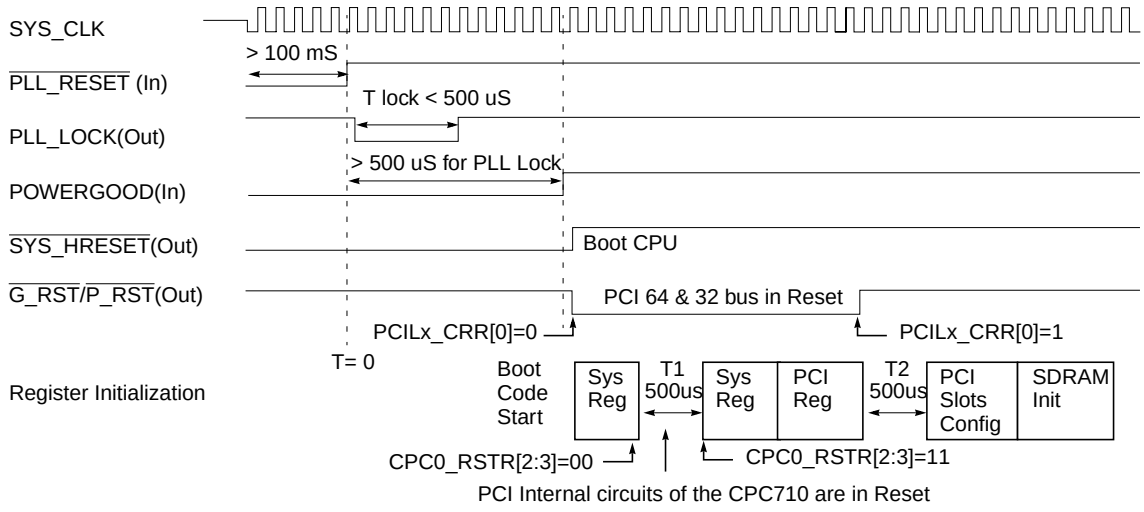


Figure 5-1. Power Up Sequence

5.2 POWERGOOD Power-On Reset

Using the system Power-On Reset POWERGOOD signal, the device resets internally and generates a reset signal to all CPUs and I/O devices. All device I/O pins go to tri-state. After a POWERGOOD

cycle, outputs on all interfaces are either floating or driven to their inactive state, except for the reset signals sent to the board as described below.

1. PowerPC bus: $\overline{\text{HRESET}}$0 and $\overline{\text{HRESET}}$1 are driven Low for the same duration as the POWERGOOD active pulse (low level).
2. PCI64 bus: $\overline{\text{G\_RST}}$ is driven Low from the beginning of the POWERGOOD assertion and remains active after POWERGOOD is deasserted. $\overline{\text{G\_RST}}$ is deactivated when the processor writes a 1 into bit 0 of “PCILx_CRR” on page 9-90 - CPC0_PCIBAR + x’000F 7EF0’ for PCI64. $\overline{\text{G\_RST}}$ is deactivated within a period that complies with the PCI Specification [2] for the 64-bit interface.
3. PCI32 bus: $\overline{\text{P\_RST}}$ is driven Low from the beginning of the POWERGOOD assertion and remains active after POWERGOOD is deasserted. $\overline{\text{P\_RST}}$ is deactivated when the processor writes a 1 into bit 0 of “PCILx_CRR” on page 9-90 - CPC0_PCIBAR + x’000F 7EF0’ for PCI32 after several PCI clocks

5.3 PLL for Clock System

Recommended PLL input controls setting:

For proper PLL operation it is recommended to set the PLL input control signals as shown in Table 5-1.

It is also recommended that the tuning bit inputs be accessible and programmable on the system board, to permit the use of other combinations in the case of difficulties to lock the PLL. This can adapted to conditions in the system environment.

Table 5-1. PLL Inputs Control Signal Setting

PLL_RANGE[1:0]	PLL Frequency range selector for the System Clock		PLL_TUNE[5:0]	Loop stability tuning control of the PLL
00	50 MHz to 100 MHz		010101	50 MHz to 100 MHz
01	58 MHz to 114 MHz		010011	58 MHz to 114 MHz
10	66 MHz to 134 MHz		010011	66 MHz to 134 MHz
11	80 MHz to 160 MHz		010011	80 MHz to 160 MHz

PLL_VDDA:

VDDA is the voltage supply pin to the analog circuits in the PLL. Noise on VDDA will cause phase jitter at the output of the PLL. To provide isolation from the noisy internal digital Vdd signal, VDDA is brought to a package pin. If little noise is expected at the board level, then VDDA can be connected directly to the digital Vdd plane. In most circumstances, however, it is prudent to place a filter circuit on VDDA as shown below. All wire lengths should be kept as short as possible to minimize coupling from other signals.

The impedance of the ferrite bead should be much greater than that of the capacitor at frequencies where noise is expected. Many applications have found that a resistor instead of a ferrite bead does a

better job of reducing jitter. The resistor should be kept to a value lower than 10 Ohms. Experimentation is the best way to determine the optimal filter design for a specific application.

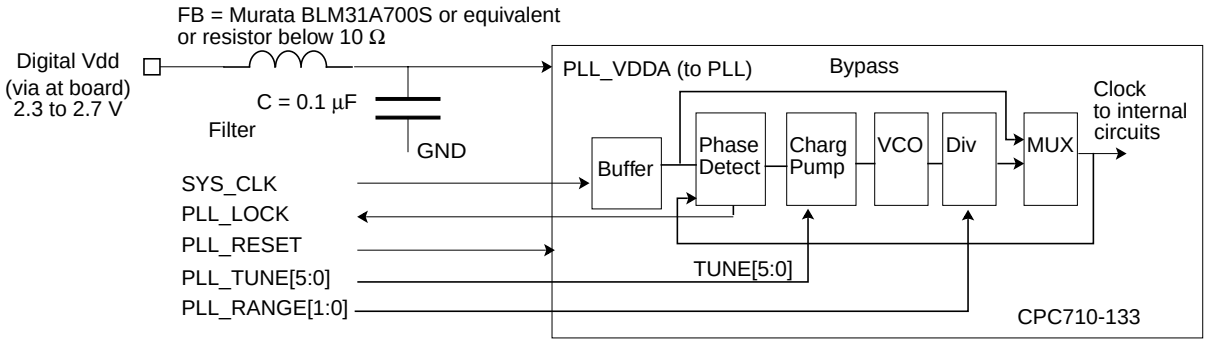


Figure 5-2. PLL Reset

The RESET signal serves two purposes. First, it holds the PLL in a reset state by forcing the VCO to operate at its minimum frequency. Second, it puts the PLL in bypass mode such that PLL Output will be buffered versions of Input SYS_Clock. RESET should be held active (high) during power-on until all of the following condition are met:

(1) All PLL inputs are stable and at their final values.
(2) SYS_CLK is stable at or below the target frequency.
(3) Any gating in the feedback path is removed.
(4) VDDA and Vdd are at their final values.

Failure to hold the PLL in reset (RESET=high) during power-on may result in VCO run-away. In this mode, output clocks are not present and the PLL can be recovered only by pulsing the RESET or VDDA pins.

5.4 Initialization of the SDRAM

To Initialize the SDRAM during the system boot it is necessary to first set all the registers of the CPC710 Memory controller and then start an automatic initialization phase by programming bit 0 of the SDRAM0_MCCR (Memory Controller Control Register).

The MRS (Mode Register Set) cycle runs only when CPC710 register SDRAM0_MCCR bit 0 is set.

In the initialization phase, the CPC710 executes the following sequence:

1. Pause 200 µs

2. Precharge-all cycle

3. 8 refresh cycles

4. MRS cycle

When the initialization phase is completed, the bit 2 of the SDRAM0_MCCR register is set to 1, such that a polling of this bit, permits to go ahead with the initialization of the others features of the CPC710.

- Examples of Programming type of SDRAM

CAS Latency = 2 or 3 The CAS Latency is set in the SDRAM Mode register according to SDRAM0_MCCR[24].

Registered mode: There is no need to program the SDRAM registered mode, because most of the SDRAM memory interface signal works with 2 CPU cycles, which permits to handle signals on cycle late. In the case of Registered SDRAM, the timings on Reading data from the SDRAM will be more critical because the data will have only one cycle to reach the Multiplexor.

5.5 Reset Individual Devices

The Connectivity Reset Register (CPC0_RSTR) at the address x'FF00 0010' provides a means to individually reset devices on the 60x bus. Bits 0 and 1 directly control $\overline{\text{SYS_HRESET}}$0 and $\overline{\text{SYS_HRESET}}$1 respectively. The remaining two bits control and Reset the PCI-32 and PCI64 internal logic of the CPC710.

The bits 0 of the CCR[0] register controls the PCI-32 or PCI64 reset signals that are outputs of the CPC710.

- PCI32 bus example:

When bit 2 of register CPC0_RSTR is asserted low, the PCI32 internal circuit of the CPC710 goes to reset.

After the reset, when the bit 2 is deasserted (Returns to high level =1) it takes 250ns before the PCI 32 bus can be used for normal accesses.

5.6 Reset in Multiprocessor mode

The sequence of Power-On Reset in Multiprocessor is the same as for a Single CPU on the 60X bus.

Simultaneously the $\overline{\text{HRESET}}$0 and $\overline{\text{HRESET}}$1 signal goes up after the POWERGOOD signal goes up. One of the two CPU get the PowerPC bus through $\overline{\text{SYS\_BR0}}$ or $\overline{\text{SYS\_BR1}}$ and get granted to access the Boot ROM at address FFFF 0100.

It can be decided for example that the CPU 0 is the Master and the CPU 1 the slave, with the CPU 0 in charge of running the code to configure the CPC710 bridge.

The Master/slave configuration is defined with the help of registers CPC0_PIDR & CPC0_RSTR

- **CPC0_PIDR Physical Identifier Register**: When BR0-BG0 signal pair is set, bit 31 is set to 0 BR1-BG1 signal pair is set, bit 31 is set to 1
- **CPC0_RSTR Connectivity Reset Register**; permit to reset CPU0 or CPU1

The first action of the boot code is to permit to the connected CPU to read the CPC0_PIDR register such that this CPU identifies if he is a Master or a slave.

In the case the CPU 1 (slave) get access first, the boot code can put him in a pooling mode until the Master complete the I/O and Memory initialization. One way is to write in the Register CPC0_RSTR.

4 Way Multiprocessor

The CPC710 handles up to 4 CPU.

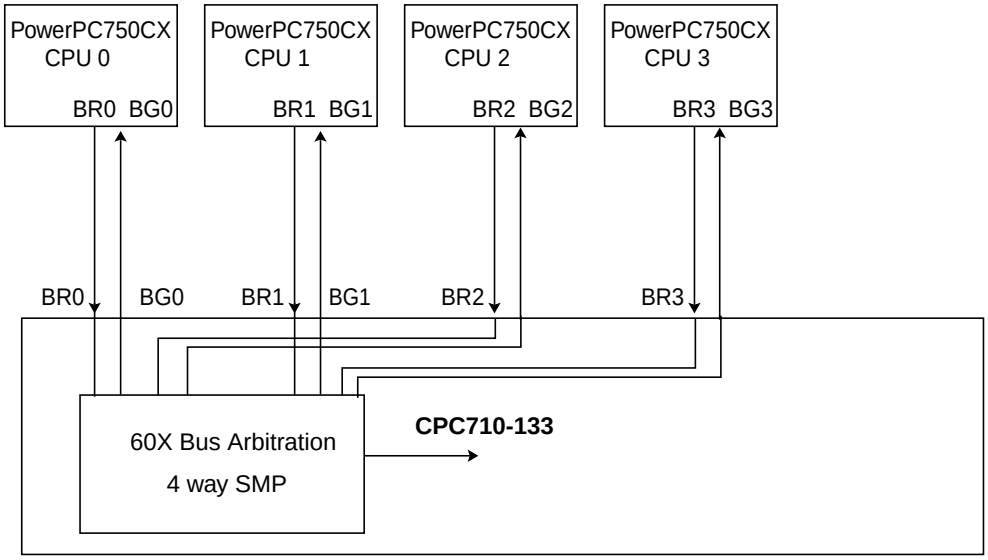


Figure 5-3. Arbitration to support of 4 Way Multiprocessing with the CPC710

5.7 Typical Register Setup Sequence

Many deviations from the proposed following example of set up are possible. However, it is important to keep the basic operations in the same sequence order as described below.

```
/*****************************************************************/
/* Typical CPC710-100 registers setup sequence (from model simulation*/
/*                                                                */
/*                              11/17/99                            */
/*****************************************************************/
/* Begin CPC710-100 registers setup sequence       */
/*================================================*/
/* 60X Interface registers setup                   */
/* ---------------------------------------------- */
CPC0_RSTR(0xff000010)  : write 0xf0000000
CPC0_UCTL(0xff001000)  : write 0x32f80000
CPC0_ABCNTL(0xff001030): write 0xb0000000
CPC0_ERRC(0xff001050)  : write 0x00c00000
CPC0_SESR(0xff001060)  : write 0x00000000
CPC0_SEAR(0xff001070)  : write 0x00000000
CPC0_PGCHP(0xff001100) : write 0x00000000
/* Memory Interface registers setup                */
/* ---------------------------------------------- */
SDRAM0_MESR(0xff001220)  : write 0x00000000
SDRAM0_MEAR(0xff001230)  : write 0x00000000
MCER0(0xff001300) : write 0x800080c0
```

```
MCER1(0xff001310) : write 0x808080c0
SDRAM0_MCCR(0xff001200)  : write 0x83b06000
/* PCI64 and PCI32 Interfaces Configuration mode setup */
/* ------------------------------------------------ */
/* Enable configuration mode for PCI64 */
CPC0_PCICNFR (0xff00000c): write 0x80000003
CPU SYNC  /* SYNC OP */
CPC0_PCIBAR (0xff200018): write 0xff400000
CPC0_PCIENB (0xff201000): write 0x80000000
/* Enable configuration mode for PCI32 */
CPC0_PCICNFR (0xff00000c) : write 0x80000002
CPU SYNC  /* SYNC OP */
CPC0_PCIBAR(0xff200018) : write 0xff500000
CPC0_PCIENB(0xff201000): write 0x80000000
/* Disable configuration modes         */
  CPC0_PCICNFR (0xff00000c) : write 0x00000000
  CPU SYNC  /* SYNC OP */


/* PCI64 Interface registers setup                  */
/* ------------------------------------------------ */

  PCILx_PIBAR(0xff4f7800) : write 0x5c000000
  PCILx_PMBAR             : write 0x5a000000
  PCILx_PR               : write 0x00008000
  PCILx_ACR              : write 0xff000000
  PCILx_MSIZE             : write 0xfc000000  /*example with  64 MB*/
  PCILx_IOSIZE            : write 0xff000000  /*example with  16 MB*/
  PCILx_SMBAR             : write 0xe0000000
  PCILx_SIBAR             : write 0x90000000
  PCILx_CSR               : write 0x00000000
  PCILx_PLSSR             : write 0x00000000

  /* PCI64 Command register setup */
  PCILx_CFGADDR(0xff4f8000)  : write 0x04000080
  PCILx_CFGDATA(0xff4f8010)  : write 0x5601


/* PCI32 Interface registers setup                  */
/* ------------------------------------------------ */

  PCILx_PIBAR(0xff5f7800) : write 0x1c000000
  PCILx_PMBAR             : write 0x1a000000
  PCILx_PR               : write 0x0000c000
  PCILx_ACR              : write 0xfe000000
  PCILx_MSIZE             : write 0xfe000000  /*example with  32 MB*/
  PCILx_IOSIZE            : write 0xff800000  /*example with   8 MB*/
  PCILx_SMBAR             : write 0xc0000000
  PCILx_SIBAR             : write 0x80000000
  PCILx_CSR               : write 0x00000000
```

```
  PCILx_PLSSR             : write 0x00000000
  PCILx_BPMDLK            : write 0xc0400000
  PCILx_TPMDLK            : write 0xc0800000
  PCILx_BIODLK            : write 0x80400000
  PCILx_TIODLK            : write 0x80800000
/* PCI32 Command register setup */
  PCILx_CFGADDR(0xff5f8000)  : write 0x04000080
  PCILx_CFGDATA(0xff5f8010)  : write 0x5601
/* ------------------------------------------------ */
/* Wait for SDRAM initialization is complete --> SDRAM0_MCCR(2) goes to a 1 */
/* Release external reset to PCI32 bus agents */
  PCILx_CRR(0xff5f7ef0) : write 0xfc000000
/* Release external reset to PCI64 bus agents */
  PCILx_CRR(0xff4f7ef0) : write 0xfc000000
/* End of CPC710-100 registers setup sequence */
/*=========================================*/
```

Chapter 6. Memory Controller

6.1 Overview

The CPC710 memory controller controls processor and I/O interactions with the memory system. The memory controller supports SDRAM and is 2-way interleaved to allow the memory to burst data on every CPU bus cycle at 133 MHz (1-1-2-1 after initial latency) using only one memory address bus. To handle critical word load, individual control of the LSB column address bits is required for the DIMM pair. This interleaved implementation allows the CPC710's memory controller to run all the memory control (except SDCKE and SDCS_) and address signals at half the System Frequency. The Data signals between the Multiplexors and the SDRAM also run at half the 60x-bus frequency. This is a main advantage for the board designer, allowing Time-Of-Fly for those signals up to 1 system cycle.

The CPC710-133 supports buffered, unbuffered and "**Registered"** SDRAM DIMMs.

The controller supports up to six dual DIMMs banks of interleaved 72-bit memory (64-bit Data + 8-bit ECC). To reduce pin count, the controller requires a Texas Instruments (TI) ALVCH162268 MUX or alternative IDT MUX to externally multiplex the 144-bit data to 72-bits for device input. Programmable parameters allow a variety of memory organizations and timings.

ECC protection is provided for all 64 bits of the data bus, detecting and correcting single- and double-bit errors.

Programmable parameters allow for a variety of memory organizations ("SDRAM Subsystem Overview" on page 6-6) and different kind of SDRAM organizations can be mixed.

The SDRAM must comply with the following requirements (compatible with the PC133 Specification [1]).

- CAS Latency = 2 or 3
- Burst length = 2
- Maximum tRCDmin allowed is 2 or 3 Clock cycles.
- Maximum tRPmin allowed is 2 or 3 Clock cycles.
- Maximum tRASmin allowed is 5 or 6 Clock cycles.

Table 6-1. Memory Performance for Cache Line Operations (ECC Active)

Pipeline Levels	Operation			CAS Latency = 2		CAS Latency = 3	
				133 MHz		133 MHz	
2	Read Burst	Initial		212 MB/s	16-1-2-1	193 MB/s	18-1-2-1
		Sustained	Page Hit	532 MB/s	4-1-2-1	426 MB/s	6-1-2-1
			Page Miss	304 MB/s	10-1-2-1	266 MB/s	12-1-2-1
	Write Burst	Initial		709 MB/s	3-1-1-1	709 MB/s	3-1-1-1
		Sustained	Page Hit	532 MB/s	5-1-1-1	532 MB/s	5-1-1-1
			Page Miss	355 MB/s	9-1-1-1	355 MB/s	9-1-1-1

6.2 Bank Definitions

The term "bank" can have several meanings, depending on context:

1. SDRAM banks
2. DIMM banks
3. Interleaved banks

6.2.1 SDRAM Banks

As shown in the following diagram, SDRAMs contain memory arranged in two or four banks. The memory controller selects these banks using bank select (BS) address pins.

6.2.2 DIMM Banks

As shown in the following diagram, DIMMs are available in single bank and dual bank configurations.

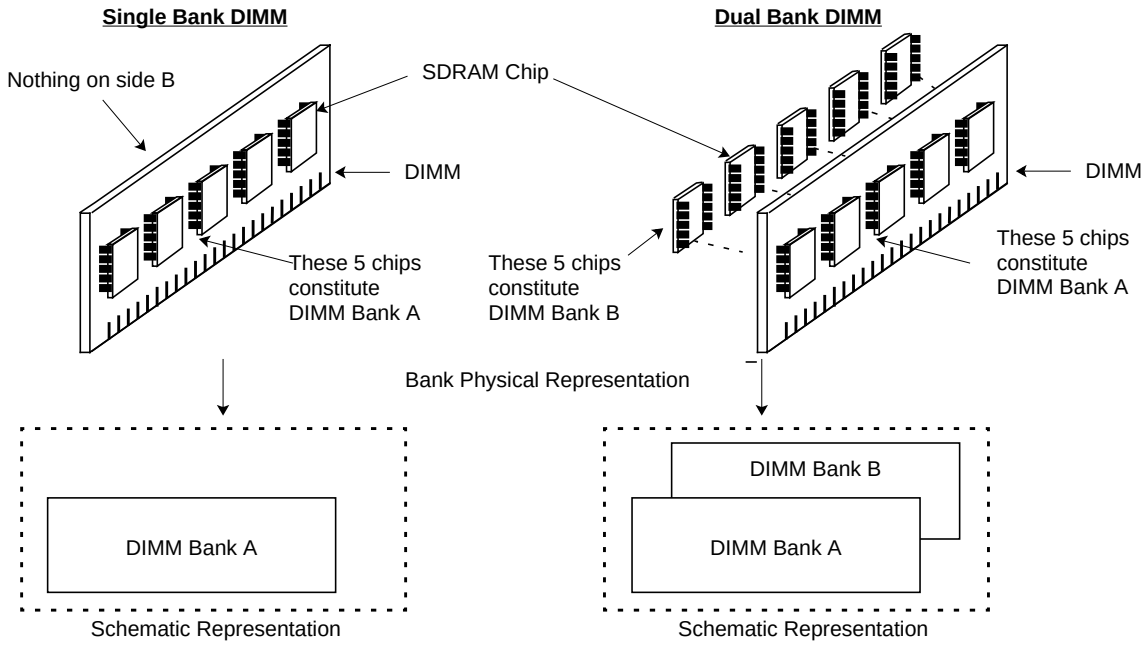
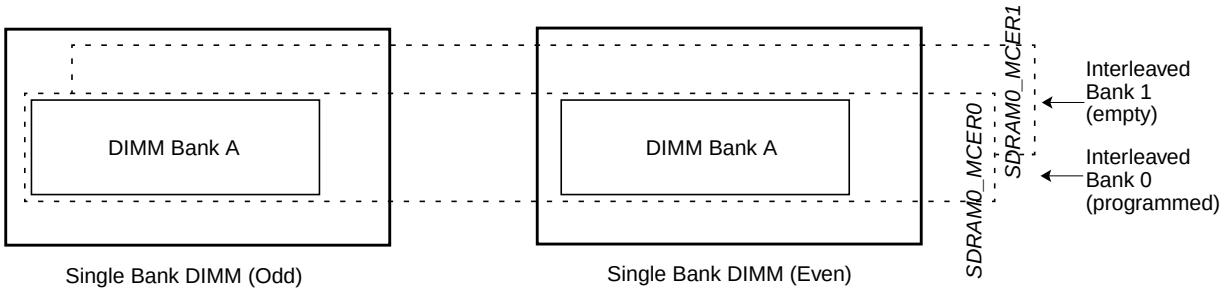


Figure 6-1. DIMM Bank Configuration

6.2.3 Interleaved Banks

An Interleaved Bank consists of two interleaved DIMM Banks. The two DIMM Banks are called Odd and Even. As shown in the following figures, SDRAM0_MCERx registers must be programmed

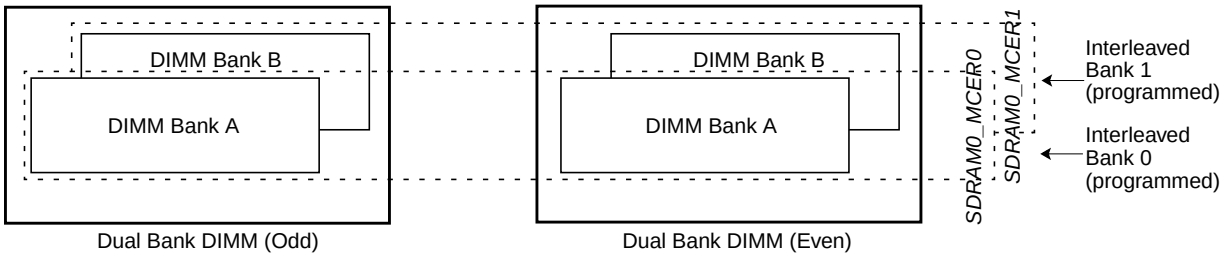
according to the DIMM Bank configuration used. Since the CPC710 works in an interleaved way, the minimum equipment required is two Single or Dual DIMM Banks.



SDRAM0_MCER0 (for Interleaved Bank 0) must be programmed.
SDRAM0_MCER1 (for Interleaved Bank 1) must be empty, or bit '0' set to '0'.

All unused SDRAM0_MCERx locations must be empty.

Figure 6-2. Programming with Single Bank DIMMs



SDRAM0_MCER0 and SDRAM0_MCER1 (for Interleaved Banks 0 and 1) must be programmed.

Figure 6-3. Programming with Dual Bank DIMMs

6.3 Memory Signal Connections

Table 6-2. SDRAM Common Signals

Signal Name	Type
MDATA[0:71]	72-bit Data
MADDR[13:1]	Address
MADDR0_ODD	
MADDR0_EVEN	

Table 6-3. External MUX Controller for Memory Data

Signal Name	Toggle for Reads	Toggle for Writes
MUX_CLKEN1B	Yes	No (’1’b)
MUX_CLKEN2B	Yes	No (’1’b)
MUX_CLKENA1	No (’1’b)	Yes
MUX_CLKENA2	No (’1’b)	Yes
MUX_OEA	No (’1’b)	No (’0’b)
MUX_OEB	No (’0’b)	No (’1’b)
MUX_SEL	High for EVEN (DH) Low for ODD (DL)	No (’1’b)

\

Table 6-4. Memory Address Bit Definition for Non-Row Column Addressing Bits

Address Bit	Definition
0-1	’00’: Base address of memory
28	Interleaving Bit 0: Even DIMMs (0, 2, 4, or 6) 1: Odd DIMMs (1, 3, 5, or 7)

Table 6-5. SDRAM Subsystem Signals

Signal Name	Type	Comments
BS0	SDRAM Bank Select	
BS1		
SDCS[0:11]	Chip Select	See the following table for connections
SDDQM[0:3)	Data Mask	up to 4 pins for load purposes (see bit 14 and 15 of SDRAM0_MCCR Register)
SDRAS[0:1]	Row Address Strobe	up to 2 pins for load purposes (see bit 14 of SDRAM0_MCCR Register)
SDCAS[0:1]	Column Address Strobe	up to 2 pins for load purposes (see bit 14 of SDRAM0_MCCR Register)
WE[0:1]	Write Enable	up to 2 pins for load purposes (see bit 14 of SDRAM0_MCCR Register)
SDCKE[0:9]	Clock Enable	10 pins for load purposes

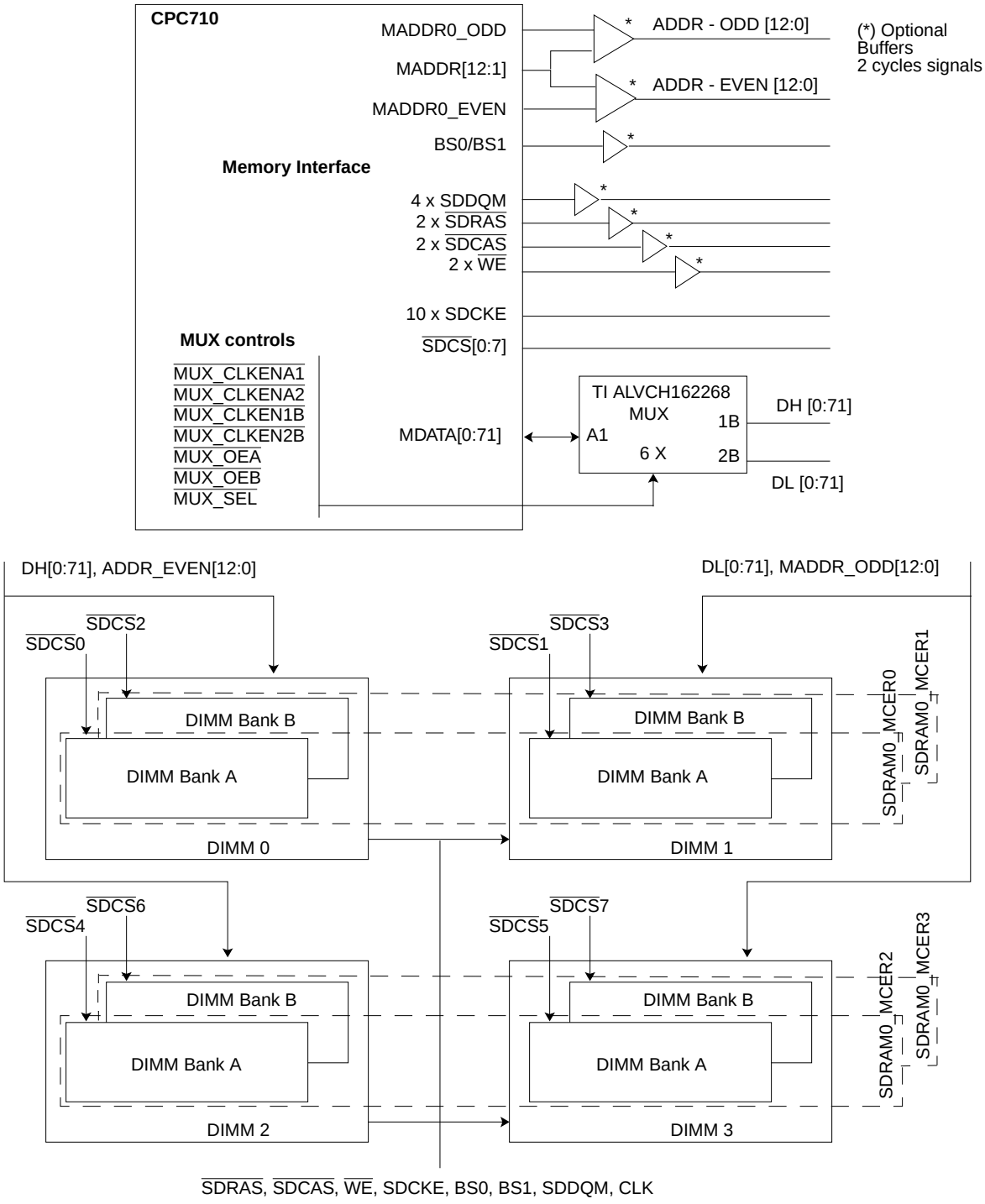
Table 6-6. SDRAM DIMM Chip Select Connections Example

Signal Name	DIMM and DIMM Bank
SDCS[0]	DIMM 0, Bank A
SDCS[1]	DIMM 1, Bank A
SDCS[2]	DIMM 0, Bank B
SDCS[3]	DIMM 1, Bank B

Table 6-6. SDRAM DIMM Chip Select Connections Example

Signal Name	DIMM and DIMM Bank
$\overline{\text{SDCS}}$[4]	DIMM 2, Bank A
$\overline{\text{SDCS}}$[5]	DIMM 3, Bank A
$\overline{\text{SDCS}}$[6]	DIMM 2, Bank B
$\overline{\text{SDCS}}$[7]	DIMM 3, Bank B

6.4 SDRAM Subsystem Overview



Note: The input clock for the SDRAM is the 60x bus clock, which is not driven by the device.

Figure 6-4. SDRAM Interface Block Diagram

6.5 Supported SDRAM Organizations

The CPC710 is fully compatible with the JEDEC Standard. The following table shows a subset of all supported SDRAMs.

Table 6-7. Supported DIMMs

DIMM Size (MByte)	SDRAM Addressing bit (Row/Col/Bank)	SDRAM (Mbits x I/Os)	Number of Chips per Bank with ECC	DIMM Size (MByte)	SDRAM Addressing bit (Row/Col/Bank)	SDRAM (MBits x I/Os)	Number of Chips per Bank with ECC
8M Single	11/8/1	1Mx16	4 + 1				
16M Single	11/9/1 11/9/1 11/8/2 12/8/1	2Mx8 2Mx32 2Mx32 2Mx8	8 + 1 2 + 1 2 + 1 8 + 1	16M Dual	11/8/1	1Mx16	4 + 1
32M Single	11/10/1 11/10/1 12/9/1 12/8/2 13/8/1	4Mx4 4Mx16 4Mx4 4Mx16 4Mx16	16 + 2 4 + 1 16 + 2 4 + 1 4 + 1	32M Dual	11/9/1 11/9/1 11/8/2 12/8/1	2Mx8 2Mx32 2Mx32 2Mx8	8 + 1 2 + 1 2 + 1 8 + 1
64M Single	12/9/2 12/9/2 13/8/2 13/9/1	8Mx8 8Mx32 8Mx32 8Mx8	8 + 1 2 + 1 2 + 1 8 + 1	64M Dual	11/10/1 11/10/1 12/8/2 12/9/1 13/8/1	4Mx4 4Mx16 4Mx16 4Mx4 4Mx16	16 + 2 4 + 1 4 + 1 16 + 2 4 + 1
128M Single	12/10/2 12/10/2 13/10/1 13/9/2	16Mx4 16Mx16 16Mx4 16Mx16	16 + 2 4 + 1 16 + 2 4 + 1	128M Dual	12/9/2 12/9/2 13/8/2 13/9/1 12/9/2	8Mx8 8Mx32 8Mx32 8Mx8 8Mx16	8 + 1 2 + 1 2 + 1 8 + 1 4 + 1
256M Single	13/10/2	32Mx8	8 + 1	256M Dual	12/10/2 12/10/2 12/10/2 13/10/1 13/9/2	16Mx4 16Mx8 16Mx16 16Mx4 16Mx16	16 + 2 8 + 1 4 + 1 16 + 2 4 + 1
512M Single	13/11/2	64Mx4	16 + 2	512M Dual	13/10/2 13/10/2	32Mx8 32Mx4	8 + 1 16 + 2
				1024M Dual	13/11/2	64Mx4	16 + 2

Note: The number of chips per SDRAM0_MCERx is double the number of chips per DIMM bank.

6.6 SDRAM Buffering Requirements

The SDRAM interface is designed to run in a 133 MHz environment. Because signal loading is critical, some outputs connect to four or eight pins. The following table lists loads and running frequencies for all SDRAM signals that use the 60x bus clock.

Table 6-8. SDRAM Input Signal Frequencies

Signal Name	Running Frequency	Maximum Input Capacitance		Note
		SDRAM (Note 1)	Unbuffered DIMM (Note 2)	
$\overline{\text{SDCS}}$	BUS_CLK	5pF	30pF	3
SDCKE	BUS_CLK	5pF	50pF	3
MDATA0/1	BUS_CLK/2	7pF	15pF	
MADDR0/1	BUS_CLK/2	5pF	50pF	
BS	BUS_CLK/2	5pF	50pF	
$\overline{\text{SDRAS}}$	BUS_CLK/2	5pF	50pF	
$\overline{\text{SDCAS}}$	BUS_CLK/2	5pF	50pF	
$\overline{\text{WE}}$	BUS_CLK/2	5pF	50pF	
SDDQM	BUS_CLK/2	5pF	50pF	

1. These are usual values for a single SDRAM chip (V=3.3V, T=25C, f=1MHz)
2. These are usual values for an unbuffered DIMM (8 x 1M x 16) (V=3.3V, T=25C, f=1MHz)
3. Signal is critical, runs at full speed.

6.7 Typical SDRAM Signals

The following figure shows all possible combinations of signals that the CPC710 can generate, which are decoded by the SDRAM.

The normal sequence of operation with SDRAM controlled by the CPC710 is:

1. Activation
2. Read or Write
3. Precharge
4. Suspend mode

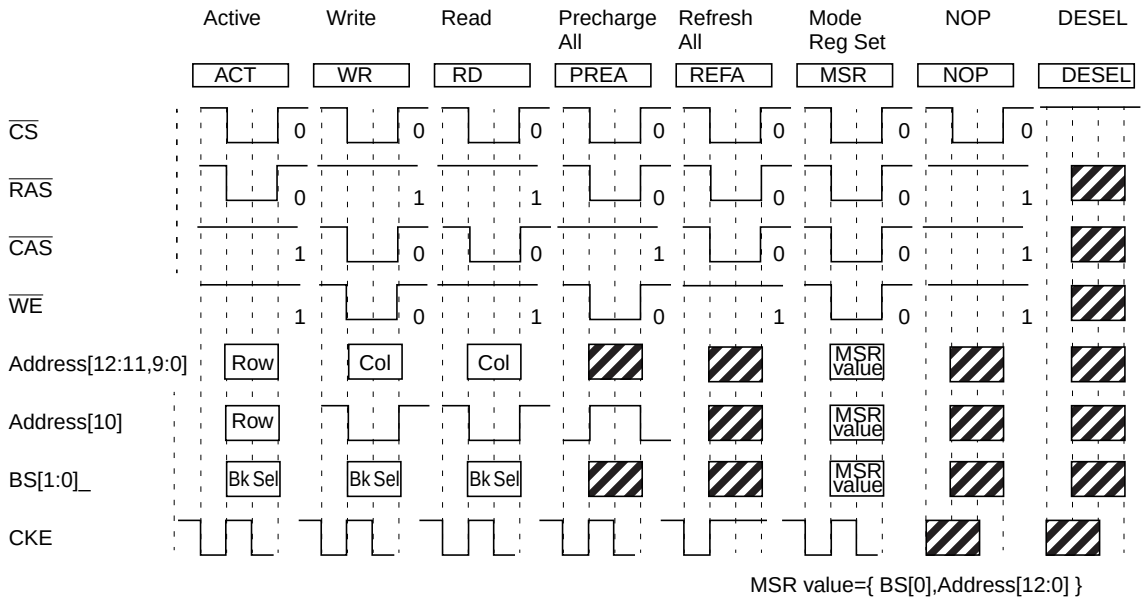


Figure 6-5. SDRAM Commands issued by the CPC710

6.7.1 CKE (Clock Enable) Signal

As shown following, the CPC710 memory controller generates signals that have to be sampled by the SDRAM Memory, based on the CKE (Clock Enable) signal.This mode of operation is fully compatible with the JEDEC Standard.

At time T0 and T2 in the following example, the CKE is sampled "ON" at the rising edge of the CLK signal, such that at the next rising edge of the CLK, the SDRAM memory is controlled. At time T1 and T3 the RAS,CAS,WE,CS combination is decoded to control the memory.

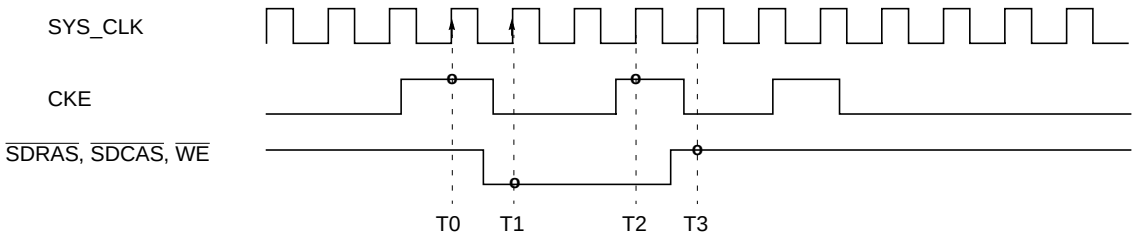


Figure 6-6. Use of the CKE signal for SDRAM Control

6.8 Mapping of System address to SDRAM Memory address with the CPC710

The following table gives the correspondence between RAS,CAS,BS address bits and PowerPC bus address bits for the supported SDRAMs by the CPC710.

Table 6-9. System Address Mapping

Msb — Address on the 60X bus

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
																													Organization			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27		Row/Col/Bank	$\overline{\text{MCERx}}$[26:29]		
								BS0	R10										R0	C7							C0	Odd/Even	11/8/1	b'1010		
							C8	BS0	R10										R0	C7							C0		11/9/1	b'0010		
							BS1	BS0	R10										R0	C7							C0		11/8/2	b'0001		
						C9	C8	BS0	R10										R0	C7							C0		11/10/1	b'0011		
							R11	BS0	R10										R0	C7							C0		12/8/1	b'1011		
						C8	R11	BS0	R10										R0	C7							C0		12/9/1	b'1100		
						BS1	R11	BS0	R10										R0	C7							C0		12/8/2	b'0100		
						R12	R11	BS0	R10										R0	C7							C0		13/8/1	b'0110		
					BS1	C8	R11	BS0	R10										R0	C7							C0		12/9/2	0 (default)		
					BS1	R12	R11	BS0	R10										R0	C7							C0		13/8/2	b'0111		
					R12	C8	R11	BS0	R10										R0	C7							C0		13/9/1	b'1000		
				C9	BS1	C8	R11	BS0	R10										R0	C7							C0		12/10/2	b'0101		
				R12	C9	C8	R11	BS0	R10										R0	C7							C0		13/10/1	b'1001		
				R12	BS1	C8	R11	BS0	R10										R0	C7							C0		13/9/2	0 (default)		
			C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		13/10/2	0 (default)		
		C11	C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		13/11/2	0 (default)		
	C12	C11	C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		13/12/2	0 (default)		
			R13	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		14/9/2	b'1101		
		R13	C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		14/10/2	b'1110		
	R13	C11	C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		14/11/2	b'1111		
R13	C12	C11	C9	R12	BS1	C8	R11	BS0	R10										R0	C7							C0		14/12/2	0 (default)		

1. The Memory Controller interleaves with only one memory address bus. To handle critical word load, individual control of the LSB column address bits is required for the DIMMs. MADDR0_ODD is used for the LSB address of the even and odd DIMMs.
2. Bit 10 is never used as address during $\overline{\text{CAS}}$ phase.

The following table is similar to the previous one, but it gives correspondence between the bit address on the SDRAM interface and the bit address on the PowerPC bus.

Table 6-10. SDRAM Address Mapping

	MADDR																BS1	BS0	Organization	
	13	12	11	10	9	8	7	6	5	4	3	2	1	0					Row/Col/Bank	$\overline{\text{MCERx}}$[26:29]
Row Address Col Address																			11/8/1	b'1010
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 **A7**	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27			**0**	A8	11/9/1	b'0010
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27			**A7**	A8	11/8/2	b'0001
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 **A6**	A11 **A7**	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27			**0**	A8	11/10/1	b'0011
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27			**0**	A8	12/8/1	b'1011
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27			**0**	A8	12/9/1	b'1100

Table 6-10. SDRAM Address Mapping

Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	**A6**	A8	12/8/2	b'0100
Row Address Col Address	A0 0	**A6** A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	**0**	A8	13/8/1	b'0110
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	12/9/2	0 (default)
Row Address Col Address	A0 0	**A6** A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	13/8/2	b'0111
Row Address Col Address	A0 0	**A5** A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	**0**	A8	13/9/1	b'1000
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 **A4**	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	12/10/2	b'0101
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 **A5**	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	**0**	A8	13/10/1	b'1001
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	13/9/2	0 (default)
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	13/10/2	0 (default)
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	13/11/2	0 (default)
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	13/12/2	0 (default)
Row Address Col Address	**A3** 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	14/9/2	b'1101
Row Address Col Address	**A2** 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	14/10/2	b'1110
Row Address Col Address	**A1** 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	14/11/2	b'1111
Row Address Col Address	A0 0	A4 A1	A7 A2	A9 -	A10 A3	A11 A6	A12 A20	A13 A21	A14 A22	A15 A23	A16 A24	A17 A25	A18 A26	A19 A27	A5	A8	14/12/2	0 (default)

1. The Memory Controller interleaves with only one memory address bus. To handle critical word load, individual control of the LSB column address bits is required for the DIMMs. MADDR0_ODD is used for the LSB address of the even and odd DIMMs.
2. Bit 10 is never used as address during $\overline{\text{CAS}}$ phase.

6.9 Memory Controller Registers

6.9.1 SDRAM0_MCCR Register

The Memory Controller Control Register contains all the parameters to fit the Memory Controller to the Synchronous DRAM components used. The following table describes how to program this register, described in "SDRAM0_MCCR" on page 9-116.

Table 6-11. SDRAM Control Register Programming

Bit(s)	Description
0	**Global System Memory Address Space Enable** 0: The CPC710 will not respond to addresses specified in Memory Configuration Extent Register (SDRAM0_MCERx) 1: System memory address space enabled.

Table 6-11. SDRAM Control Register Programming

Bit(s)	Description
1	**Diagnostic Mode** This bit is used to control presentation of double-bit ECC errors to the system. This bit is primarily intended for use in memory testing at power on time. Software can use this bit when testing memory and or ECC logic in order to avoid the hardware generating a machine check for double-bit ECC errors. The error however, is still logged into the SDRAM0_MEAR 0: Normal Mode: Multi-bit ECC error will generate Machine Check 1: Diagnostic Mode: Multi-bit ECC does NOT generate Machine Check; logged in SDRAM0_MEAR & SDRAM0_MESR
2	**SDRAM Initialization Status (read-only)** 0: SDRAM initialization is not completed. 1: SDRAM initialization is completed.
3 - 4	**ECC Mode** This field provides software with a means to control ECC generation and checking. b'01' is provided to allow software direct read/write access to the ECC byte that is associated with every doubleword of data stored in memory and also provide a mechanism to verify the memory controller's ECC generation and checking logic. In this mode, byte lane 0 data (MSB of a double word) is written to the ECC byte instead of the normal ECC code byte. Data byte 0 will be forced to all zeros. For reads, byte 0 will contain the byte stored in the ECC byte, not the data at byte 0. ECC checking is not enabled for reads in this mode. This mode also allows firmware write single-bit and multi-bit errors into memory to allow for ECC logic testing. 00: Normal generation and checking of ECC codes. The CPC710 will generate the normal ECC code when writing to memory and check ECC when reading. 01: ECC check disabled; Byte lane 0 routed to/from ECC check field. Data byte 0 forced to all zeros This mode is provided to allow software direct read/write access to the ECC byte that is associated with every doubleword of data stored in memory and also provide a mechanism to verify the memory controller's ECC generation and checking logic. In this mode, byte lane 0 data (MSB of a double word) is written to the ECC byte instead of the normal ECC code byte. Data byte 0 will be forced to all zeros. For reads, byte 0 will contain the byte stored in the ECC byte, not the data at byte 0. ECC checking is not enabled for reads in this mode. This mode also allows firmware write single-bit and multi-bit errors into memory to allow for ECC logic testing. The CPC710 will still generate normal ECC codes when writing to memory. 10: ECC check disabled; Normal routing of data and normal ECC code generation The CPC710 will still generate normal ECC codes when writing to memory. 11: Reserved
5 - 7	**Row Cycle Time for SDRAM Auto-refresh (t_{RC})** Allows to fit the delay between the Refresh Command and the next Activation. This delay has to be at least the tRC-min value specified in the SDRAM data sheet. 000: 5 bus cycles 001: 6 bus cycles 010: 7 bus cycles 011: 8 bus cycles 100: 9 bus cycles 101: 10 bus cycles 110: 11 bus cycles 111: 12 bus cycles
8	**Size Code Encoding Type For Bank** Select the encoding code for Size Bank defined in SDRAM0_MCERx[16:25] bit field 1: Normal mode (4 MB to 1GB Size available) 0: Extended mode (4 MB to 4GB Size available)
9	**Reserved**. Must be Left to 0
10	**Data Pacing Mode** (Must be set to 1 for SDRAM)
11	**Chip Select duplicated Mode 1** 1: SDCS_[0:3] signals are using SDCS_[4:7] outputs by Multiplexing
12	**Chip Select duplicated Mode 2** 1: SDCS_[0:3] signals are using SDCS_[8:11] outputs by Multiplexing
13	**Reserved.** Must be set to 0

Table 6-11. SDRAM Control Register Programming

Bit(s)	Description
14	**First Multiplexing Control of SDRAM Signals SDQM** 1: SDQM signals are using SDRAS1_, SDCAS1_ and WE1_ outputs by Multiplexing
15	**Second Multiplexing Control of SDRAM Signals SDQM** 1: SDQM signals are using PCG_ARB outputs by Multiplexing
16-17	**SDRAM Type** 00: Standard modules SDRAM 01: Registred DIMMs SDRAM mode 2 (not available: for future use) 10: Registred DIMMs SDRAM mode 1 11: Reserved (unpredictable result)
18	**Reserved.** Must be left to 0
19	**Registered DIMMs extra clock cycle** 1: Following signals are shifted by one Clock cycle MUX_CLKEN1B_ MUX_CLKEN2B_ MUX_SEL_
20	**Reserved.** Must be set to 1
21	**Registered DIMMs Write extra cycle:** 1: Data to be written to the SDRAM Memory is maintained one more cycle
22	**Registered DIMMs extra clock cycle** 1: Following signals are internal shifted by one Clock cycle (SDCKE and SDCS signals must be external shifted by one Clock cycle) MUX_CLKENA2_ MUX_OEB_ SDRAS_0 SDRAS_1 SDCAS_0 SDCAS_1 WE_0 WE_1 MADDR0_ODD MADDR0_EVEN MADDR1-13 BS0 and BS1
23	**Reserved.** Must be left to 0
24	**Extend CAS Latency (CL)** 0: CAS Latency programmed to 2 cycles. 1: CAS Latency programmed to 3 cycles.
25	**Extend RAS Precharge (tRP)** 0: RAS Precharge programmed to 2 cycles. 1: RAS Precharge programmed to 3 cycles.
26	**Extend RAS-to-CAS Delay (tRCD min)** 0: RAS-to-CAS Delay programmed to 2 cycles. 1: RAS-to-CAS Delay programmed to 3 cycles.
27	**Extend RAS Active Pulse Width (tRAS min)** 0: Programmed to 5 cycles. 1: Programmed to 6 cycles.
28	**Mutibanking Enable** 0: Multibanking Not active. 1: Multibanking is active.

Table 6-11. SDRAM Control Register Programming

Bit(s)	Description
29	**Shifted Refresh Cycles** In normal operation, the refresh of all the populated DIMMs is performed at the same time. If the 8 DIMMs are fully populated, this could produce a high current load (all SDCS_ activated at the same time). Setting bit 29 to 1, enables the controller to perform successive Refresh (only 2 SDCS_ activated at the same time); in return the refresh cycle last longer depending on the number of populated DIMMs. 0: All banks are refreshed in the same cycle. 1: Banks are refreshed one after one.
30	**Disable Page Mode** 0: Memory controller will perform fast page accesses for back to back operations if appropriate 1: Memory controller will perform fast page access only within a burst operation. It will NOT perform fast page accesses for back to back bursts even if they occur to the same $\overline{RAS}$ page.
31	**Disable Queue Same Page Override** 0: Memory queue ordering can be overridden if an operation is to the same page. 1: Memory queue always processed in order received.

6.9.2 SDRAM0_MCERx Registers

The Memory Configuration Extent Registers (SDRAM0_MCER0:5) program the start address and size of each bank. The following table shows the relationship between the DIMMs and the SDRAM0_MCERx registers.

Table 6-12. SDRAM0_MCERx to Program Functions of DIMMs

Bank Definition (DIMMs equipped)	Corresponding SDRAM0_MCERx	Note
DIMM0-Bank1 and DIMM1-Bank1	SDRAM0_MCER0	
DIMM0-Bank2 and DIMM1-Bank2	SDRAM0_MCER1	
DIMM2-Bank1 and DIMM3-Bank1	SDRAM0_MCER2	
DIMM2-Bank2 and DIMM3-Bank2	SDRAM0_MCER3	
DIMM4-Bank1 and DIMM5-Bank1	SDRAM0_MCER4	1
DIMM4-Bank2 and DIMM5-Bank2	SDRAM0_MCER5	1

1. When using SDRAM and Data Mask, Mode is active (see SDRAM0_MCCR, bit 11) and the CPC710 can support only up to four bank.

To configure contiguous address spaces with different bank sizes, software must put the largest bank sizes at the lowest addresses and continue in order to the smallest bank sizes.
To set up the SDRAM0_MCERx registers, software must read the PD bits and the ID bits for each

DIMM. These bits are located in the System I/O registers (see “SDRAM0_MCER0:5” on page 9-120). The following table describes how to initialize these registers.

Table 6-13. SDRAM0_MCERx Register Initialization

DIMM Description		Device Bank Size (MB)		If SDRAM0_MCCR[8] = 1		If SDRAM0_MCCR[8] = 0	
DIMM Size (MB)	Number of Banks per DIMM	Bank x	Bank x+1	MCER(x) Bits[16:25]	MCER(x+1) Bits[16:25]	MCER(x) Bits[16:25]	MCER(x+1) Bits[16:25]
2	1	4	Not equipped	x’3F3	off	x’3FF	off
4	2	4	4	x’3F3	x’3F3	x’3FF	x’3FF
4	1	8	Not equipped	x’3E3	off	x’3FE	off
8	2	8	8	x’3E3	x’3E3	x’3FE	x’3FE
8	1	16	Not equipped	x’3C3	off	x’3FC	off
16	2	16	16	x’3C3	x’3C3	x’3FC	x’3FC
16	1	32	Not equipped	x’383	off	x’3F8	off
32	2	32	32	x’383	x’383	x’3F8	x’3F8
32	1	64	Not equipped	x’303	off	x’3F4	off
64	2	64	64	x’303	x’303	x’3F4	x’3F4
64	1	128	Not equipped	x’203	off	x’3F0	off
128	2	128	128	x’203	x’203	x’3F0	x’3F0
128	1	256	Not equipped	x’003	off	x’3C0	off
256	2	256	256	x’003	x’003	x’3C0	x’3C0
256	1	512	Not equipped	x’002	off	x’380	off
512	2	512	512	x’002	x’002	x’380	x’380
512	1	1024	Not equipped	x’000	off	x’300	off
1024	2	1024	1024	x’000	x’000	x’300	x’300
1024	1	2048	Not equipped			x’200	off
2048	2	2048	2048			x’200	x’200
2048	1	4096	Not equipped			x’000	off

1. "DIMM size" is the size in MB of one DIMM (including Bank A and Bank B if dual bank DIMM).
2. "Number of banks per DIMM": One for single bank DIMM and two for dual bank DIMM
3. x in SDRAM0_MCER(x) = 0, 2, 4, or 6
4. a setting of “off” indicates that the bank must be disabled by setting SDRAM0_MCER(x) Bit 0 = 0.

6.10 Error Handling

The memory controller detects four errors:

1. Single-bit ECC
2. Multi-bit ECC
3. Invalid address

4. Overlapping memory extents

Errors 2, 3, and 4 are considered hard errors. If one occurs, it is logged into SDRAM0_MESR and SDRAM0_MEAR and cannot be overwritten with a subsequent hard error. Single-bit ECC errors are considered soft and once logged into the SDRAM0_MEAR and SDRAM0_MESR, can be overwritten with a subsequent hard error.

6.10.1 Single-Bit ECC Error, General Case

The hardware procedure for this error is:

1. Set the single-bit error bit in the SDRAM0_MESR register.
2. If neither a double-bit error nor an address error is present, store the syndrome in the SDRAM0_MESR and the address in SDRAM0_MEAR.
3. Corrected data is not written back to memory but forwarded to the requesting logic.
4. When Chip Programmability Register (CPC0_PGCHP) bit 17 = 1, a Machine Check is performed to signal the processor that it could rewrite correct data to memory.

Software must write zeros to the SDRAM0_MESR to clear this error. If more than one single-bit ECC error occurs before the SDRAM0_MESR clears, only the first error is recorded. When a double-bit ECC error or an address error occurs, the software overwrites the SDRAM0_MESR and SDRAM0_MEAR.

6.10.2 Single-Bit ECC Error, Special Case

For non-burst write transactions that do not span an entire aligned double-word, the Memory Controller performs a read-modify-write sequence to memory. If the read portion of the sequence results in a single-bit ECC error, the error is not logged into the SDRAM0_MESR and SDRAM0_MEAR for both the diagnostic and normal modes. However, the memory controller automatically writes corrected data to memory.

6.10.3 Invalid Address Error

An Invalid Address error is detected by the Memory Controller when an address does not match one of the eight configuration extents. The hardware procedure for this error is:

1. If no hard errors are in the SDRAM0_MESR register, set the invalid address error bit.
2. If no hard errors are in the SDRAM0_MEAR register, store the address.
3. In diagnostic mode, the Memory Controller responds with dummy data and indicates an Invalid Address error to the requesting logic. To enable further error logging, the software writes zeros into the SDRAM0_MESR.

When more than one address error occurs before the SDRAM0_MESR clears, only the first error is recorded. No Single- or Double-Bit ECC errors are logged into the SDRAM0_MESR and SDRAM0_MEAR if they occur after the Invalid Address error.

6.10.4 Double-Bit ECC Error, General Case

The hardware procedure for this error is:

1. Set the Double-Bit error bit in SDRAM0_MESR if no hard errors are present.
2. Store the syndrome in SDRAM0_MESR and the address in SDRAM0_MEAR if no hard errors are present.
3. In normal mode, indicate the error to the requesting logic with DAT_ERR for the appropriate double word that failed in memory.
4. In diagnostic mode, do not indicate Double-Bit errors with DAT_ERR.
5. Software must write zeros to the SDRAM0_MESR to clear errors.
6. If more than one Double-Bit error occurs before the SDRAM0_MESR clears, only the first error is recorded.
7. A Single-Bit error is not logged into SDRAM0_MESR and SDRAM0_MEAR when it occurs after a Double-Bit error.
8. If an Address error occurs after a Double-Bit error, it is not be logged into SDRAM0_MESR and SDRAM0_MEAR.

6.10.5 Double-Bit ECC Error, Special Case

For non-burst write transactions that do not span an entire aligned double-word, the Memory Controller performs a read-modify-write sequence to memory. If the read portion of this sequence results in a Double-Bit error, the controller indicates the error to the requesting logic through the response bus instead of using DAT_ERR. If this occurs in diagnostic mode, the error is logged into SDRAM0_MESR and SDRAM0_MEAR, but not reported through the response bus.

6.10.6 Overlapping Memory Extents

Overlapping Memory Extents are not detected until an access occurs to an address mapped to two different configuration extents. When an overlap condition is detected, the hardware follows the following procedure:

1. Set the Overlapping Memory Extent error bit in SDRAM0_MESR if no hard errors exist.
2. Store the address in SDRAM0_MEAR if no hard errors exist.
3. The Memory Controller responds with dummy data for reads, ignores write data, and indicates an Invalid Address error to the requesting logic. To enable further error logging, the software writes zeros into the SDRAM0_MESR.

When a Single-bit or a hard error occurs after an Overlapping Memory Extent, the error is not logged into SDRAM0_MESR and SDRAM0_MEAR.

6.10.7 ECC Check-bit and Syndrome

The Error Checking and Correction (ECC) provides double-bit error detection, and single-bit error correction for a 64-bit double Word. When a double word is stored in memory, and 8 bit ECC check code is generated by the CPC710 and stored with the data.

Therefore, the data width on the memory bus is 72 bit (64 bit data and 8 ECC check bits).

6.10.7.1 Single-Bit Error Correction

When a double word is read from memory, a new ECC code is generated and compared in the CPC710 to the one that was stored in memory. If the two codes are different, an error exists in either the data or the ECC word.

The difference in these ECC codes is called the *Syndrome*.

If the Syndrome is zero, there is obviously no error. Otherwise there is some type of error. The Syndrome can be decoded to determine which bit in the word or in the ECC code is incorrect, providing it is a single-bit error, such that it can be immediately repaired by logical inversion.

Even multi bit errors can be detected; however, only single-bit errors can be corrected. Table 6-14 below shows the Error Correcting Code check-bit and Syndrome matrix for single-bit errors, as implemented with the CPC710 memory bus controller.

Table 6-14. ECC Check-Bit /Single-Bit Error Syndrome Matrix

Byte	0	1	2	3	4	6	6	7	Chk bits
Data	00000000 01234567	00111111 89012345	11112222 67890123	22222233 45678901	33333333 23456789	44444444 01234567	44555555 89012345	55556666 67890123	012345 67
S0	11111111	001..110	.10.1.01	1..1....	...1....	...1..11	.0.111..	111.0...	1.....
S1	111.00.0	11111111	001..110	.10.1.01	1..1....	...1....	...1..11	.0.111.0	.1.....
S2	000111..	111.00.0	11111111	001.0110	.10.1.01	1..1....	...1....	...1..11	..1.....
S3	...1..11	.00111..	111.00..	11111111	001..110	.10.1.01	1.01....	...1....	...1....
S4	.001....	...1..11	.00111.0	111.00.0	11111111	001..110	.10.1.01	1.01.0..	....1...
S5	1..1....	...1....	...1..11	.00111..	111.00..	11111111	001..110	.10.1.01	.....1..
S6	.1..1.01	1..1....	...1....	...1..11	.00111.0	111.00.0	11111111	001..110	......1.
S7	..1..110	.10.1.01	1..1....	...1....	...1..11	.00111..	111.00.0	11111111	1

Check bits are defined in the table as the XOR in one row of the active “1’s” in the 64 bits of data:

- The single bit error Syndrome for each bit to be repaired by inversion in the CPC710 can be read in each column of the table. For example, Bit 16 is defective if syndrome is S0:7 = x011xxx1.
- The other possible nonzero Syndrome values not in the table indicates multiple-bit errors.

Since a multi-bit error could cause the same Syndrome bit to be set (B'1')as a single-bit error, further qualifiying is necessary. This is done by requiring B'0's in certain bit positions to distinguish between single-bit from multiple-bit errors. (a full decode of the 8 syndromes bits is not necessary). For example, a single-bit error has occurred for Data bit zero, if syndrome bits 0,1 and 5 are B'1''s and bit 2 is a B'0'. Even further qualifying of the syndromes may be necessary.

6.10.7.2 Additionnal Information for Software

The following is a table of data values required to produce certain ECC check bits. This is used by software for testing purpose. For each of the desired check bits, the specified data is just one of the possible value.

Table 6-15. Data Values Required for Check Bits

Desired Check Bits (C0-C7)	Possible Data Value
x'55	'x'0832 C000 0000 0000'
x'AA	'x'B2A0 0000 0000 0000'
x'OO	'x'6300 0000 0000 0000'
x'O1	'x'2880 0000 0000 0000'
x'02	'x'4880 0000 0000 0000'
x'04	'x'DF5C 0000 0000 0000'
x'08	'x'BAA0 0000 0000 0000'
x'10	'x'357C 0000 0000 0000'
x'20	'x'08A0 0000 0000 0000'
x'40	‘x'40A0 0000 0000 0000'
x'80	'x'4820 0000 0000 0000'

Chapter 7. PCI Bridges

The CPC710 PCI bridges execute load and store operations from the CPU to the PCI buses. It also provides an interface for PCI devices to access system memory. The PCI Bridge logic fully supports the PCI Local Bus Specification [2]. The following table describes the physical connections for PCI devices on the PCI32 bus in a desktop system.

Table 7-1. PCI32 Bus Device Physical Connection Example

Device	ARB Level	RESET Signal	IDSEL Signal
PCI SLOT 0	$\overline{\text{P\_REQ0}}$/$\overline{\text{P\_GNT0}}$	$\overline{\text{P\_RST}}$	IDSEL1
PCI SLOT 1	$\overline{\text{P\_REQ1}}$/$\overline{\text{P\_GNT1}}$	$\overline{\text{P\_RST}}$	IDSEL2
PCI SLOT 2	$\overline{\text{P\_REQ2}}$/$\overline{\text{P\_GNT2}}$	$\overline{\text{P\_RST}}$	IDSEL3
ETHERNET CHIP	$\overline{\text{P\_REQ3}}$/$\overline{\text{P\_GNT3}}$	$\overline{\text{P\_RST}}$	IDSEL4
SCSI CHIP	$\overline{\text{P\_REQ4}}$/$\overline{\text{P\_GNT4}}$	$\overline{\text{P\_RST}}$	IDSEL5
ISA BRIDGE CHIP	$\overline{\text{P\_REQ5}}$/$\overline{\text{P\_GNT5}}$	$\overline{\text{P\_RST}}$	IDSEL6
MPIC	Not required	POWERGOOD	IDSEL7

7.1 Address Map

The two PCI bus bridges in the CPC710 both implement the register maps listed in the following table. The PCI Host Bridge Standard configuration space is Little Endian.

Table 7-2. PCI Bus Bridge Configuration Address Map

Area	Real Address	Name	Use	Page
System Standard Configuration Space	x'FF20 0000'		Reserved	
	x'FF20 0004'		Reserved	
	x'FF20 0018'	CPCO_PCIBAR	Base Address Reg. for Bridge Registers	9-23
	x'FF20 001C' to x'0FFF'		Reserved	
Device Specific Configuration Space	x'FF20 1000'	CPC0_PCIENB	PCI BAR Enable Register	9-25
	x'FF20 1004' to x'1FFF'		Reserved	

1. Read Only Register, write is ignored

7.2 System Standard Configuration Registers

System Standard Configuration Registers can only be accessed with 60x bus configuration cycles directed to a specific PCI bridge. Both of the CPC710 PCI bridges must be configured before any PCI configuration cycles can be issued. The registers provide a mechanism for firmware to assign a 1MB

address space in the system memory map for the location of the PCI bridge facilities (PCP0_PCIBAR register). For detailed descriptions of these registers, refer to the following:

- “CPC0_PCIBAR” on page 9-23
- “CPC0_ABCNTL” on page 9-10

7.3 System PHB Registers

The PCI bridge logic follows the PowerPC PCI Host Bridge (PHB) Architecture, including the enhanced error detection and error reporting features. The logic deviates from PHB Architecture only in its ability to recover from PCI errors.

7.4 PCI Bus Commands

The following table describes the subset of PCI bus commands supported by the CPC710.

Table 7-3. Supported PCI Commands

C/BE[3:0]	Command	Support as Initiator	Support as Target
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle	Yes	No
0010	I/O Read Cycle	Yes	No
0011	I/O Write Cycle	Yes	No
0100	Reserved		
0101	Reserved		
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved		
1001	Reserved		
1010	Configuration Read	Yes	Yes (PCI64 only)
1011	Configuration Write	Yes	Yes (PCI64 only)
1100	Memory Read Multiple	No	Yes
1101	Dual Address Cycle	No	No
1110	Memory Read Line	Yes	Yes
1111	Memory Write and Invalidate	Yes	Yes

7.4.1 PCI Master Memory Read Cycles

When the CPC710 receives a memory read bus cycle from system memory, it first initiates a CLEAN cache operation to the processor bus. Processor accesses to this cache line are $\overline{\text{SYS_ARTRY}}$ed until the memory read is finished. If the cache line is determined to be stale in memory, the PCI bus cycle

is retried. The following figure shows the states the CPC710 follows when executing a PCI memory read cycle.

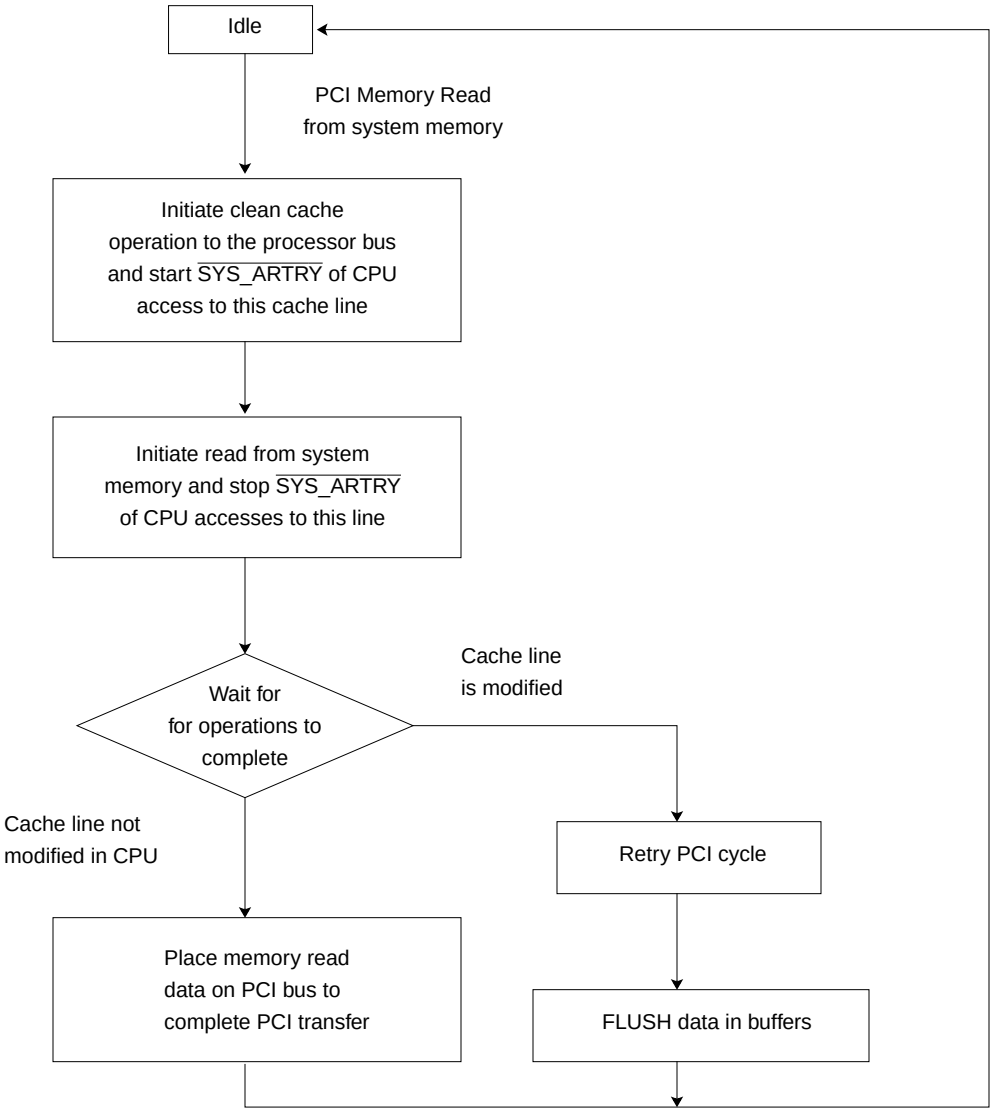


Figure 7-1. PCI Memory Read State Diagram

7.4.2 PCI Master Memory Write Cycles

When the CPC710 receives a memory write bus cycle to system memory, it first initiates a FLUSH cache operation to the processor bus. Processor accesses to this cache line will be SYS_ARTRYed until the memory write is finished. If the cache line is determined to be stale in memory, the PCI bus

cycle is retried. The following sequence describes the states the CPC710 follows when executing a PCI memory write cycle.

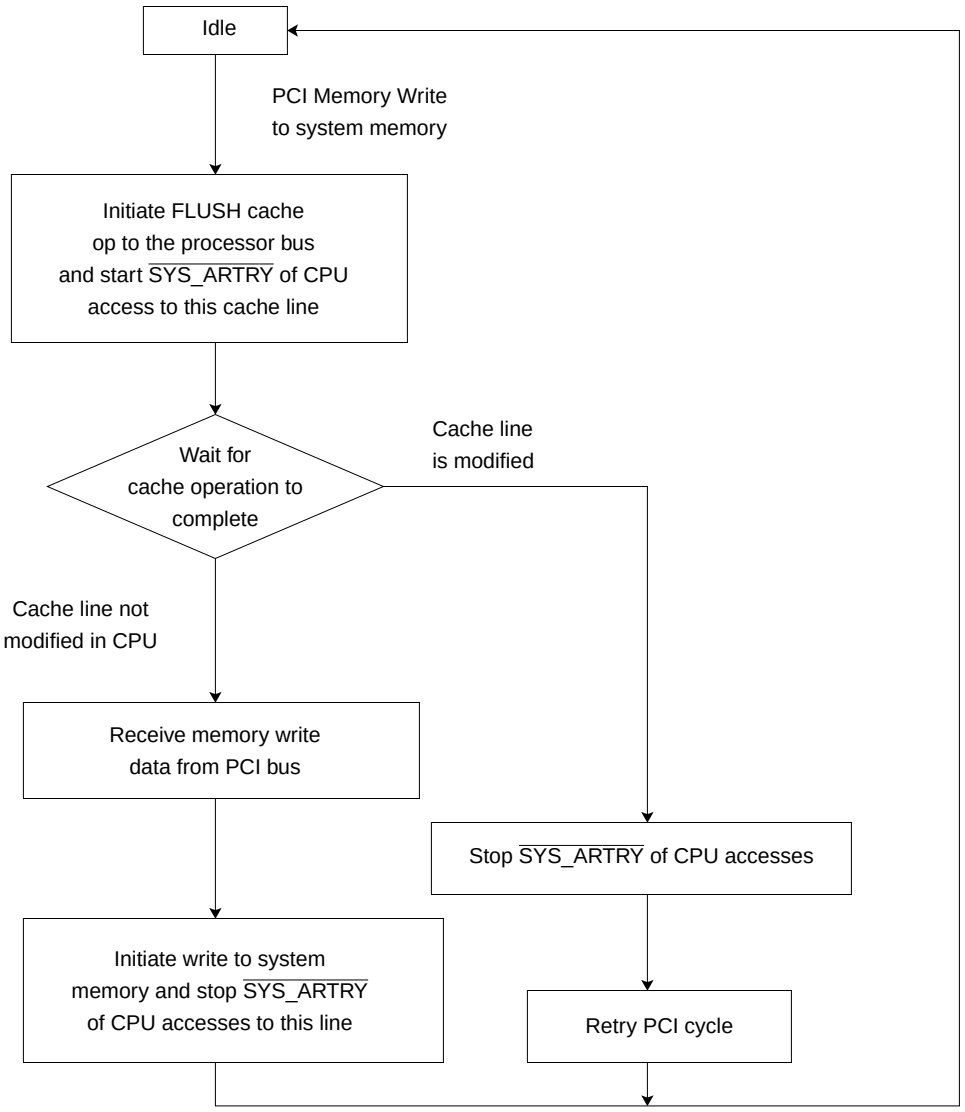


Figure 7-2. PCI Memory Write State Diagram

7.4.3 Configuration Cycles

The CPC710 implements **Configuration Mechanism #1** as specified in the PCI Local Bus Specification [2]. This mechanism uses an indirect addressing model with the PCILx_CFGADDR and PCILx_CFGDATA registers. The configuration target address is first written into PCILx_CFGADDR and then an access is made to PCILx_CFGDATA to generate a configuration transfer. Each PCI bridge has a separate set of these registers. When each decodes an access to its PCILx_CFGDATA register, it performs different operations depending on the values stored in PCILx_CFGADDR.

Table 7-4. PCI Configuration Cycle Matrix

CONFIG_ADDRESS Register Fields					Action	Notes
Enable	Bus#	Device#	Function#	Register#		
0	x	x	x	x	Configuration not enabled. Returns 0's on loads and ignores write data.	1
1	BUS# < PCICx_BUSNO	x	x	x	Invalid Bus# in PCILx_CFGADDR. Returns 1's on loads and ignores store data. No access made to PCI Bus.	1, 2
	BUS# = PCICx_BUSNO	0	0	x	Access to PCI Bridge configuration space. Read/Write to PCI Bridge configuration registers.	1, 2
		1-21	x	x	Configuration access to device on PCI Bus. TYPE 0 configuration cycle on PCI bus.	
		22-30	x	x	Not supported. TYPE 0 configuration cycle with no IDSELs on. Returns 1s on loads and ignores store data.	
		31	7	0	Special cycle command. Special cycle command issued to PCI Bus.	
	BUS# > PCICx_BUSNO BUS# < PCICx_SUBNO	x	x	x	Configuration access to bridge on PCI Bus. TYPE 1 configuration cycle on PCI Bus	1, 2, 3
	BUS# > PCICx_BUSNO BUS# > PCICx_SUBNO	x	x	x	Invalid bus# PCILx_CFGADDR. Returns 1s on loads and ignores store data. No access made to PCI Bus.	

1. Firmware must insure the SUBNO register in the PCI header is greater than or equal to the PCICx_BUSNO register in the PCI header. Unpredictable results can occur if this is not true.
2. The PCI Bridge performs a compare of the BUS NUMBER field in the PCILx_CFGADDR register and the BUS NUMBER field in the bridge's 256-byte PCI header.
3. The PCI Bridge performs a compare of the BUS NUMBER field in the PCILx_CFGADDR register and the SUBORDINATE BUS NUMBER field in the bridge's 256-byte PCI header.

If there is no response to a configuration cycle (no DEVSEL# detected), the CPC710 Master-Aborts the cycle, sets the Master Abort bit in the PCI Status register, and completes the processor cycle normally by returning all ones on reads and ignoring data on writes.

7.4.3.1 Type 0 Configuration Cycles

During a TYPE 0 configuration cycles, the CPC710 provides on the AD[31-11] lines the IDSEL of the device to be configured on the PCI32 or PCI64 bus as described in the PCI 2.1 specification.

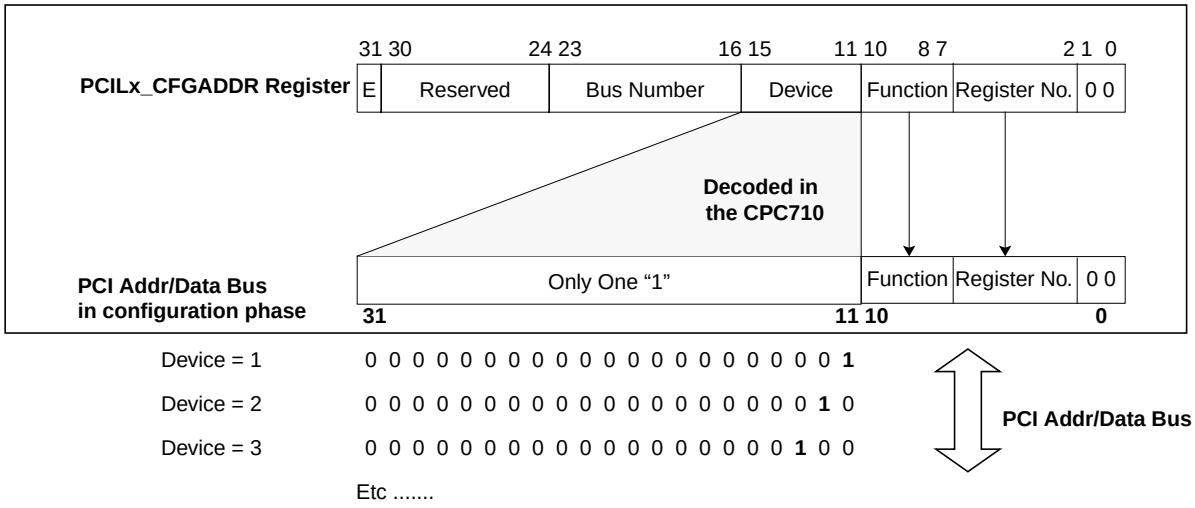

31 30
24 23
16 15
11 10
8 7
2 1 0
PCILx_CFGADDR Register
E
Reserved
Bus Number
Device
Function
Register No.
0 0
Decoded in the CPC710
PCI Addr/Data Bus in configuration phase
Only One “1”
Function
Register No.
0 0
31
11 10
0
Device = 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
Device = 2
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
Device = 3
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
Etc .....
PCI Addr/Data Bus

Figure 7-3. PCI Address/Data Bus for Type 0 Configuration Cycles

7.4.3.2 Type 1 Configuration Cycles

For Type 1 configuration cycles, the CPC710 directly copies the contents of the PCILx_CFGADDR register to the Address/Data signals on the PCI bus. However, Address/Data[1:0] contains ’01’ to indicate a Type 1 configuration cycle.

7.5 PCI Performance Estimates

Table 7-5. PCI to Memory Sustained Throughput

	Read	Write	Units
PCI64bit @ 66 MHz	378	348	MByte/sec.
	16-1-1-1-1-1-1-1-3-1.....PRAL	12-1-1-1-1-1-1-1-5-1.....PRAL	66MHz PCI Cycles

Assumptions:

- 4KBytes Burst
- PCI Master parked on PCI bus
- No other activity present
- Adapter supports fast back-back transfers for stores to memory
- No L1 or L2 cache hits

PARL (PCI Rearbitration Access Latency) min = 1 cycle

Table 7-5. PCI to Memory Sustained Throughput

	Read	Write	Units
PCI32bit @ 33 MHz	130	130	MByte/sec.
	11-1-1-1-1-1-1-1-....-1-PRAL	11-1-1-1-1-1-1-1-....-1-PRAL	33MHz PCI Cycles
Assumptions: • 4KBytes Burst • PCI Master parked on PCI bus • No other activity present • Adapter supports fast back-back transfers for stores to memory • No L1 or L2 cache hits PARL (PCI Rearbitration Access Latency) min = 1 cycle			

Table 7-6. CPU to PCI Sustained Throughput

Operation	Loads @ 100 MHz Bus		Stores @ 100 MHz Bus		Units
	PCI32bit @ 33 MHz	PCI64bit @ 66 MHz	PCI32bit @ 33 MHz	PCI64bit @ 66 MHz	
Burst 32 bytes	71	194	71	194	MB/s
Single 8 bytes	30	67	30	67	MB/s
Single 4 bytes	17	33	17	33	MB/s
Assumptions: • CPU is parked on 60x bus • 1 Level Pipeline • CPC710 parked on PCI bus • No other activity present					

7.6 PCI Master Error Handling

For PCI bus errors detected on CPU initiated transfers, refer to “Error Handling for CPU-Initiated Transactions” on page 4-15. The following table describes the error handling performed for PCI master errors.

Table 7-7. PCI Master Error Handling (Page 1 of 2)

Operation	Error	Mode	Action	Notes
Any PCI Bus Transfer	Address Parity Error	Enabled by PCI PCICx_CMD register bit 6	Save encoded arb level in PCILx_CSR register Set Address parity error detected bit in PCILx_CSR register Set Parity error detected bit 15 in PCI status register Place PCI address in PCILx_PSEA register Activate SERR signal if enabled by bit 8 PCI PCICx_CMD register Set Signalled SERR bit in PCI Status register if enabled Target abort PCI transfer if address matches Set Signaled target abort bit in PCI status register Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
		Disabled	Set Parity error detected bit 15 in PCI status register Complete PCI transfer normally if address matches	1
	Detected SERR Active PCI Bridge Logic Idle		Set Detected SERR active bit in PCILx_CSR register Save encoded ARB level in PCILx_CSR register Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
Access to System Memory	Single Bit Error		Set single-bit error and syndrome in SDRAM0_MESR Set error address in SDRAM0_MEAR Return corrected data to PCI device Proceed normally with PCI transaction	1
	Double Bit Error	Normal	Set double-bit error in SDRAM0_MESR Set error address in SDRAM0_MEAR Set memory error bit in PCILx_CSR register In the case of double-bit ECC error after a PCI to Memory access, either signal $\overline{\text{SYS\_MCP0}}$ or $\overline{\text{SYSMCP1}}$ will be active, depending on the setting of PCILx_PR[3]. 0: $\overline{\text{SYS\_MCP0}}$ signals Machine Check 1: $\overline{\text{SYS\_MCP1}}$ signals Machine Check Loads: - Target abort PCI transfer - Set signaled target abort bit in PCI status register - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Stores: - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
		Diagnostic	Set double-bit error in SDRAM0_MESR Set error address in SDRAM0_MEAR Return uncorrected data to PCI device Proceed normally with PCI transaction	1

Notes:
1. “Normally” means that dummy zeros are returned for loads and write data is ignored.

Table 7-7. PCI Master Error Handling (Page 2 of 2)

Operation	Error	Mode	Action	Notes
Access to System Memory (cont'd)	Invalid Address		Set invalid address error in SDRAM0_MESR Set error address in SDRAM0_MEAR Set invalid memory address bit in PCILx_CSR Loads: - Target abort PCI transfer - Set signaled target abort bit in PCI status register - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Stores: - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
	Detected SERR Active		Set SERR detected error bit in PCILx_CSR register Save encoded ARB level in PCILx_CSR register Target abort PCI transfer Set signaled target abort bit in PCI status register Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
	Detected PCI Bus Data Parity Error during PCI Master Store	Enabled by PCI PCICx_CMD register bit 6	Activate the PERR signal Set parity error bit 15 in PCI Status register Complete PCI transfer, however, Flush store data; do not write to memory	
		Disabled	Set parity error bit 15 in PCI Status register Proceed normally with PCI transaction	1
	Detected PERR during PCI Master Load		Proceed normally with PCI transaction	1
	Received Master Abort		Proceed normally with PCI transaction	1
	PCI Bus Timeout: IRDY Count Expired		Target abort PCI transfer Set signaled target abort bit in PCI status register Set PCI bus time-out error in PCILx_CSR register Save encoded ARB level in PCILx_CSR register Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	
Access to Device on 2nd PCI Bus	Internal Response Bus Contains "PCI Error" Status		2nd PCI bridge logs errors same as CPU initiated 2nd PCI bridge does NOT drive $\overline{\text{SYS\_MCP0:1}}$ pin Set PCI - PCI error bit in PCILx_CSR register Save encoded ARB level in PCILx_CSR register Save PCI address in PCILx_PSEA register Loads: - Target abort PCI transfer - Set signaled target abort bit in PCI status register - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$ Stores: - Signal Machine Check with $\overline{\text{SYS\_MCP0:1}}$	

Notes:

1. "Normally" means that dummy zeros are returned for loads and write data is ignored.

Chapter 8. DMA Controller

8.1 Introduction

The data transfers between the system memory and the PCI buses can be performed either by the DMA controller or by a PCI master on one of the PCI buses which can access in Read or Write the System memory (See the PCI section).The DMA is initiated either by a PowerPC specific instruction or by writing to the DMA0_XTARx register. The DMA is defined with one channel, and with several type of mode of operations. To signal the end of the DMA operation, the External interrupt $\overline{IT2}$ is raised.

8.2 Mode of operation of the DMA

A complete DMA transfer can be done in the following modes that can be programmed in the DMA Global Control Register (DMA0_GSCRx):

- Elementary
- Extended Mode
- Chained
- Skip of Cache line

The DMA controller runs with an elementary block of up to 4 KB of data to transfer.

In the Extended Mode, an automatic address increment is performed at the end of each elementary DMA transfers. Up to 65,000 iterations (or loops) of elementary DMA's can be programmed with address increments to transfer up to 256 MB of data in a single DMA. The end-of-transfer DMA interrupt $\overline{IT2}$ is raised only after completion of the multiple elementary DMAs loops.

The chained DMA permits with one single command to have several DMA with different lengths different starting and target addresses, operations to be performed in a single operation. To have a chained DMA, the bit 31 of the DMA Transfer Write Back Address Register (DMA0_XWARx) have to set to 1 to enable the chaining at the completion of the first DMA operation. The chained DMA descriptors are stored in the memory cache line at the address defined in the DMA0_XWARx.

The Skip of Cache line mode is a way to fill table with one elementary cache line out of "n" in a DMA operation. The Cache line increment is defined in the DMA0_XCLRx register.

8.3 Starting the DMA

- Write in the DMA0_XTARx register

The write in the DMA0_XTARx register results in the start of a DMA operation.

- eciwx or ecowx instruction

DMAs are initiated by either a eciwx (read: Data from Memory to PCI) or ecowx (write: PCI to Memory) instruction from the processor and ended by an External Interrupt command. The controller uses an elementary burst of 32 Bytes on the PCI bus to facilitate interleaved PCI bus operations. The eciwx and ecowx instructions use the processor's internal address translation logic to present real

addresses on the system bus. This eliminates the need for external hardware to translate virtual addresses and for software to calculate real addresses. Because the DMA is virtual, no software overhead is required for pinning system memory that would otherwise be needed if the DMA operated in real address mode.

Execution of an eciwx or ecowx instruction involves the same sequence as a normal cache inhibited load and store with a few exceptions. The processor calculates an effective address, translates it, and presents the resulting real address to the system bus as normal. However, this address bus does not select the slave. The address is passed to the slave to be used on a subsequent transfer. The slave is selected by a 4-bit Resource ID (PCICx_REVID) that is placed on the $\overline{\text{SYS\_TBST}}$ and SYS_TSIZ[0:2] signals by the processor.

The device is selected for these transactions when the PCICx_REVID on the bus matches Configuration Register bits 8-11 in the device's System Control Register. The bus transaction is always a single beat regardless of the $\overline{\text{SYS\_TBST}}$ signal setting. While the DMA is occurring, the device monitors the bus for a TLB Sync (resulting from normal page maintenance by the OS kernel) to terminate the transfer. Software can then restart the transfer at the faulting address.

The DMA Controller transfers data between system memory and PCI only. It cannot perform memory-to-memory transfers. DMA operation is transparent to the PCI adapter, which behaves as a PIO slave device. Although eciwx and ecowx both initiate DMA, the preferred instruction is ecowx because it writes to the system bus. eciwx is provided to avoid access violation errors on pages marked read-only.

Software ensures proper implementation of the DMA operation, including address alignments and page boundaries. The device aborts a DMA transfer when any of the following conditions are detected:

- TLBSYNC operation detected (internal commands are completed before termination).
- Improper DMA transfer setup.
- Second DMA transfer initiated when one is already in progress.
- The transfer crosses a page boundary.

8.4 DMA Transfer Registers

Several registers support the DMA transfer process. They are mapped to two different address spaces so the software can mark the x'FF1C xxxx' range as user space and the 'FF1E xxxx' range as privileged space. This provides protection needed to allow the eciwx and ecowx instructions to be executed by application level software. The registers are listed in the following table and are described in "DMA Registers" on page 9-45.

Table 8-1. DMA Transfer Register Summary

Register	User			Privileged			Description
	Address	Bits	Mode	Address	Bits	Mode	
DMA0_GSCRx	FF1C 0020	[0:31]	R	FF1E 0020	[0:31]	R/W	Global Control Register
DMA0_XCLRx	FF1C 0030	[0:31]	R	FF1E 0030	[0:31]	R	Cache line increment Register

Table 8-1. DMA Transfer Register Summary

Register	User			Privileged			Description
	Address	Bits	Mode	Address	Bits	Mode	
DMA0_XSCRx	FF1C 0040	[0:31]	R/W	FF1E 0040	[0:31]	R/W	DMA Transfer Control Register
DMA0_XSSRx	FF1C 0050	[0:31]	R	FF1E 0050	[0:31]	R	DMA Transfer Status Register
DMA0_XPARx	FF1C 0070	[0:3]	R	FF1E 0070	[0:3]	R/W	PCI Address Register
		[4:31]	R/W		[4:31]	R	
DMA0_XWARx	FF1C 0090	[0:31]	R	FF1E 0090	[0:31]	R/W	Writeback Address Register
DMA0_XTARx	FF1C 00A0	[0:31]	R	FF1E 00A0	[0:31]	R	Translated Address Register

The steps for executing a DMA transfer with software are:

1. Initialize DMA0_XSCRx to indicate length and direction of transfer.
2. Initialize DMA0_XPARx with the PCI address. The PCI logic takes the address in the DMA0_XPARx register and applies the translation as described in “CPU to PCI Addressing Model” on page 2-2.
3. Initialize DMA0_XWARx with the address to which the device writes to indicate status following transfer.
4. Clear cache line status in memory at address specified in DMA0_XWARx.
5. Execute the ecowx instruction (or eciwx if read only page) to start transfer.
6. Wait until an End-of-DMA transfer interrupt (IT) occurs, then read the status on the memory address specified in XWAR. Reset bit 4 of the DMA0_GSCRx register to acknowledge the IT. Alternatively, perform cache polling to the memory address specified in DMA0_XWARx and wait until the cache status flag changes from x’00’ to x’FF’.

8.4.1 DMA Transfer Status Cache Line Descriptor for Chained DMA’s

The following table shows the definition of the 64-bit of status stored in main memory at the address defined by the DMA0_XWARx register. Only bits 32-63 of the second double-word of the Write Back Status cache line are valid. All other bytes in the cache line must be ignored.

Table 8-2. DMA Transfer Status Cache Line Definition

Bit(s)	Description
Status Double-word 0	
0-63	Reserved
Status Double-word 1	
0-63	Undefined
Status Double-word 2	
0-63	Undefined
Status Double-word 3	
0-31	x’0000 0000’

Table 8-2. DMA Transfer Status Cache Line Definition

Bit(s)	Description
32 - 39	Poll Status Cache Line Valid Flag x'00' - Initial value set by software. Indicates status cache line is not valid. x'FF' - Written by hardware to indicate that the status cache line has been updated and is valid
40	Transfer Complete 0 - Transfer is not complete 1 - Transfer is complete
41	TLBSYNC Detected 0 - No TLBSYNC Detected 1 - TLBSYNC detected during DMA transfer Transfer
42	Reserved
43	Page Crossing Error 1 - Page Crossing detected during DMA transfer
44	Second DMA Transfer Halt 1 - DMA transfer operation in progress was halted due to start of second DMA transfer operation
45	Unaligned ecowx/eciwx Address 1 - Address associated with ECOWX/ECIWX is not word aligned
46	Unaligned Transfer Error 1 - Address alignment error
47	Address Increment Alignment Error 1 - Improper alignment of addresses when Address Increment bit is off
48	Invalid PCI Address 1- DMA0_XPARx did not match any PCI extents
49 - 50	Reserved
51 - 63	Transfer Length This field contains the number of bytes remaining when the transfer was completed or aborted

8.5 DMA Procedure

The DMA transfer process begins when the 60x logic detects an ecowx or eciwx transaction on the processor bus. If the PCICx_REVID bits in the CPC710 and the CPC0_UCTL register match the PCICx_REVID bits on the $\overline{\text{SYS\_TBST}}$ and SYS_TSIZ[0:2] lines, the 60x logic accepts the transfer. If the instruction is an ecowx, the 60x logic $\overline{\text{SYS_TA}}$s the bus for dummy write data and sends a DMA Transfer Write command to the DMA Controller.

The internal address bus associated with the Transfer Write command contains the address from the processor bus. This address is placed in the DMA0_XTARx register by the DMA Controller. During the processor address tenure, the 60x logic sets an internal flag to indicate special handling of TLBSYNC operations on the processor bus. If the flag is not set, the 60x logic ignores all TLBSYNC operations on the processor bus. If the flag is set, a TLBSYNC operation on the bus causes the 60x logic to place a one cycle pulse on the UX6_TLB_SYNC line to the DMA Controller. The 60x logic continuously $\overline{\text{SYS_ARTRY}}$s the TLBSYNC bus operation until it receives a one cycle pulse on the internal

UXI_XFER_DONE line from the DMA Controller. This pulse also resets the 60x logic's internal flag to perform special handling of the TLBSYNC operations.

Note: Since the PowerPC601 processor does not issue TLBSYNC operations, the 60x logic must treat any SYNCs following a TLBI as a TLBSYNC operation when operating with a PowerPC601 processor.

When the eciwx instruction is used, the 60x logic performs the same steps except that the 60x logic internally sends a DMA Transfer Read command to the DMA Controller and waits for a dummy read data response. The dummy read data is then placed on the processor bus to complete the eciwx transfer on the processor bus. The internal flag for special handling of TLBSYNC is set during the eciwx address bus tenure on the processor bus.

After the DMA Controller receives the DMA Transfer command, it issues a Load Pointer command on the internal command bus to the appropriate PCI bus bridge logic unit. This transfers the address in DMA0_XPARx to the PCI bus bridge pointer register. The DMA Controller then issues a series of Blit commands, or internal Elementary Commands from the DMA Controller to the PCI logic, to the same PCI bus bridge logic unit that transfers the data. The first Blit command contains the memory address stored in the DMA0_XTARx register.

The PCI bus bridge logic receives the Blit commands and then executes the transfer. For Blit Reads, the DMA Controller first determines whether the read from memory requires a snoop transaction. If the read is coherent, the controller issues a snoop command to the 60x logic. If the snoop fails, the controller retries the snoop until it passes. Once the snoop passes, a Blit Read command is transmitted to the PCI bus bridge logic. The PCI Bridge logic executes the command and then increments the value in its pointer register by the size of the transfer unless the Address Increment field in the Load Pointer command is set to No Increment. Blit Write commands are handled in same way except the transfer is from I/O to System Memory.

Note: The DMA Controller should wait a minimum of eight cycles before reissuing snoop commands after a snoop fail response.

After the transfer is complete, the controller signals the 60x logic by activating UXI_XFER_DONE for one cycle. The controller then issues a Write with Kill to the address specified in DMA0_XWARx register to indicate to software that the transfer is complete. The controller issues a Kill Cache to the 60x logic, and upon receiving a clean response, issues a Write command to system memory. The write to memory need only be a single beat write to the bytes reserved for DMA transfer status.

8.5.1 Special Boundary Conditions

Due to queueing in the 60x logic, a pulse could be placed on the TLBSYNC line to the DMA Controller before the controller receives an ecowx or eciwx. In this case, the controller waits until it receives an ecowx or eciwx and then immediately terminates the DMA transfer. When two DMA transfers overlap, the controller ignores the TLBSYNC pulse if a DMA transfer is nearly complete. However, because the 60x logic could have an eciwx or ecowx queued, the controller would have to remember the TLBSYNC pulse to terminate the second DMA transfer properly. To do this, the 60x logic indicates the presence of an eciwx or ecowx instruction in its queue to the controller.

Chapter 9. Register Summary

The registers for the CPC710 are specified in three regions. Except for the Standard PCI Configuration Space, which uses indirect addressing, all the registers can be defined in the upper 16 MB of the 4 GB address range.

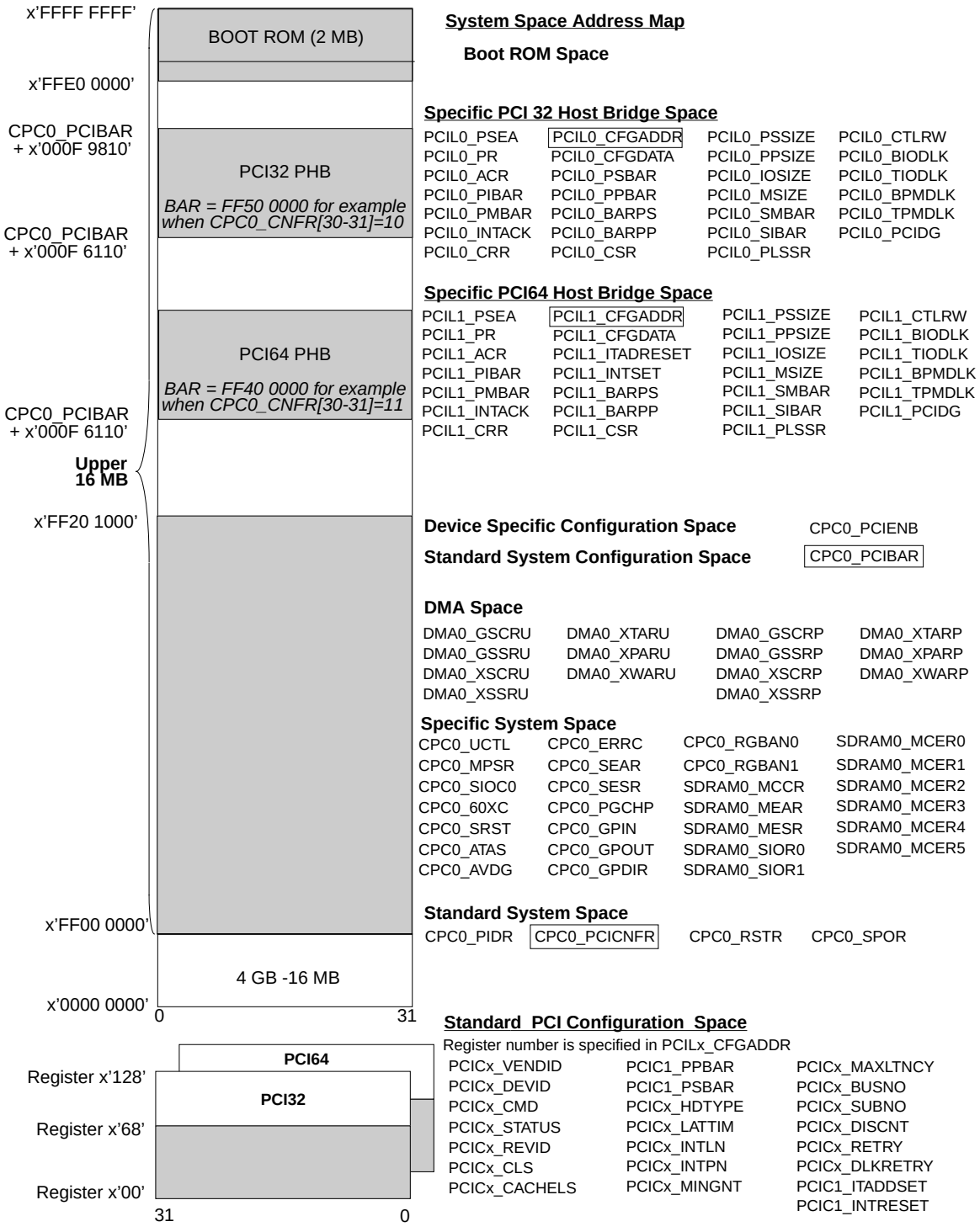
Each of these registers is described in “Alphabetical List of Registers” on page 9-9.

9.1 System Register Space

The upper 16 MB of the 4 GB address range (0xFF00 0000 to 0xFFFF FFFF) is reserved for system support functions. Table 9-1 on page 9-3 describes the System Space Registers supported. These registers are defined as Big Endian unless otherwise noted. If the processor is operating in Little Endian mode, software must issue Load & Store reverse instructions to access these registers.

The CPC710 responds to all addresses listed in Table 9-1 on page 9-3 with a minimum granularity of 4K blocks. Accesses to these registers must be single word accesses on word boundaries or unpredictable results may occur.

Shaded address ranges indicate areas where CPC710 will respond with TEA (addressing error is detected and logged in the System Error Status Register (CPC0_SESR x’FF00 1060 bit 15 or bit 22).



Note: BOXED registers are key registers that define PCI bus configuration and register settings.

Figure 9-1. CPC710 Register Address Map

Table 9-1. System Registers List

Address	Name	Use	Page	Notes
x'FF00 0000' to x'FF00 0007'		Reserved		
Standard System Registers				
x'FF00 0008'	CPC0_PIDR	Physical Identifier Register	9-29	1
x'FF00 000C'	CPC0_PCICNFR	Connectivity Configuration Register	9-24	see p 33
x'FF00 0010'	CPC0_RSTR	Connectivity Reset Register	9-32	
x'FF00 0020'	CPC0_RTBR	Refresh Time Base Register	9-33	
x'FF00 00E8'	CPC0_SPOR	Software POR Register	9-41	7
Specific System Registers				
x'FF00 1000'	CPC0_UCTL	Universal Control Register	9-43	
x'FF00 1010'	CPC0_MPSR	Multi-Processor Semaphore Register	9-22	
x'FF00 1020'	CPC0_SIOC0	System I/O Control 0	9-38	
x'FF00 1030'	CPC0_ABCNTL	60x Arbiter Control Register	9-10	
x'FF00 1040'	CPC0_SRST	CPU Soft Reset Register	9-42	7
x'FF00 1050'	CPC0_ERRC	Error Control Register	9-17	
x'FF00 1060'	CPC0_SESR	System Error Status Register	9-35	
x'FF00 1070'	CPC0_SEAR	System Error Address Register	9-34	
x'FF00 1080'		Reserved		
x'FF00 1090'	CPC0_SIOC1	System I/O Control 1	9-40	
x'FF00 1100'	CPC0_PGCHP	Chip program Register	9-26	
x'FF00 1110'	CPC0_RGBAN0	Free Register 0	9-30	
x'FF00 1120'	CPC0_RGBAN1	Free Register 1	9-31	
x'FF00 1130'	CPC0_GPDIR	GPIO Direction Register	9-19	
x'FF00 1140'	CPC0_GPIN	GPIO Input Register	9-20	
x'FF00 1150'	CPC0_GPOUT	GPIO Output Register	9-21	
x'FF00 1160'	CPC0_ATAS	Address Transfer Attribute for Snoop Reg	9-13	
x'FF00 1170'	CPC0_AVDG	Device Diagnostic Register	9-15	
x'FF00 1174' to x'FF00 11FF'		Reserved		
x'FF00 1200'	SDRAM0_MCCR	Memory Controller Control Register	9-116	

1. RO: Read Only Register
2. All bits can be read. Only bits [4:31] can be written
3. All bits can be read. Only bits [0:3] can be written
4. Four beat burst read operations allowed to this address space; Single byte writes only
5. Not decoded by system logic
6. Byte accesses allowed
7. WO: Write Only Register
8. Range that IBM Dual Bridge and Memory Controller responds to is programmable

Table 9-1. System Registers List

Address	Name	Use	Page	Notes
x'FF00 1210'	SDRAM0_MWPR	Memory Write Protect Register	9-125	
x'FF00 1220'	SDRAM0_MESR	Memory Error Status Register	9-124	
x'FF00 1230'	SDRAM0_MEAR	Memory Error Address Register	9-123	
x'FF00 1300'	SDRAM0_MCER0	Memory Configuration Extent Register 0	9-120	
x'FF00 1310'	SDRAM0_MCER1	Memory Configuration Extent Register 1	9-120	
x'FF00 1320'	SDRAM0_MCER2	Memory Configuration Extent Register 2	9-120	
x'FF00 1330'	SDRAM0_MCER3	Memory Configuration Extent Register 3	9-120	
x'FF00 1340'	SDRAM0_MCER4	Memory Configuration Extent Register 4	9-120	
x'FF00 1350'	SDRAM0_MCER5	Memory Configuration Extent Register 5	9-120	
x'FF00 1360'		Reserved		
x'FF00 1370'		Reserved		
x'FF00 1400'	SDRAM0_SIOR0	SIO Register 0 (DIMM PDs)	9-127	1
x'FF00 1410'		Reserved		
x'FF00 1420'	SDRAM0_SIOR1	SIO Register 1 (Planar, DIMM, CPU, etc.)	9-128	1
x'FF00 1424 to x'FF00 1FFF'		Reserved		
x'FF00 2000 to 'FF17 FFFF'		Reserved		
DMA Registers: User Privilege				
x'FF18 0000 to x'FF1C 001F'		Reserved		
x'FF1C 0020'	DMA0_GSCRU	Global Control Register (user)	9-45	1
x'FF1C 0030'	DMA0_XCLRU	DMA Cache Line Increment Register (user)	9-47	
x'FF1C 0040'	DMA0_XSCRU	Transfer Control Register (user)	9-49	
x'FF1C 0050'	DMA0_XSSRU	Transfer Status Register (user)	9-50	1
x'FF1C 0070'	DMA0_XPARU	Transfer PCI Address Register (user)	9-48	2
x'FF1C 0090'	DMA0_XWARU	Transfer Write Back Address Register (user)	9-53	1
x'FF1C 00A0'	DMA0_XTARU	Transfer Translated Address Register (user)	9-52	1
x'FF1E 0020'	DMA0_GSCRP	Global Control Register (priv)	9-45	
x'FF1E 0030'	DMA0_XCLRP	DMA Cache Line Increment Register (priv)	9-47	
x'FF1E 0040'	DMA0_XSCRP	Transfer Control Register (priv)	9-49	

1. RO: Read Only Register
2. All bits can be read. Only bits [4:31] can be written
3. All bits can be read. Only bits [0:3] can be written
4. Four beat burst read operations allowed to this address space; Single byte writes only
5. Not decoded by system logic
6. Byte accesses allowed
7. WO: Write Only Register
8. Range that IBM Dual Bridge and Memory Controller responds to is programmable

Table 9-1. System Registers List

Address	Name	Use	Page	Notes
x'FF1E 0050'	DMA0_XSSRP	Transfer Status Register (priv)	9-50	1
x'FF1E 0070'	DMA0_XPARP	Transfer PCI Address Register (priv)	9-48	3
x'FF1E 0090'	DMA0_XWARP	Transfer Write Back Address Register (priv)	9-53	
x'FF1E 00A0'	DMA0_XTARP	Transfer Translated Address Register (priv)	9-52	1
x'FF1E 00A4' to x'FF1F FFFF'		Reserved		
System Standard Configuration Registers				
x'FF20 0000'		Reserved		
x'FF20 0004'		Reserved		
x'FF20 0008' to x'FF20 0014'	Reserved			
x'FF20 0018'	CPC0_PCIBAR	Base Address Register	9-23	see p. 33
x'FF20 0020' to x'FF20 0FFF'	Reserved			
Device Specific Configuration Space				
x'FF20 1000'	CPC0_PCIENB	PCI BAR Enable Register	9-25	
x'FF20 1004' to x'FFDF FFFF'		Reserved		
BOOT ROM				
x'FFE0 0000' to x'FFFF FFFF'	IPLROM	FLASH ROM: Up to 2 MB		4, 6, 8

1. RO: Read Only Register
2. All bits can be read. Only bits [4:31] can be written
3. All bits can be read. Only bits [0:3] can be written
4. Four beat burst read operations allowed to this address space; Single byte writes only
5. Not decoded by system logic
6. Byte accesses allowed
7. WO: Write Only Register
8. Range that IBM Dual Bridge and Memory Controller responds to is programmable

9.2 Standard PCI Configuration Space

There are two sets of PCI Configuration Space registers; one for each PCI bridge. These registers are accessed by a R/W of the PCILx_CFGDATA with the address of the target register of the

corresponding PCI bus in the PCILx_CFGADDR[7:2] (Configuration Address register) which specifies the register number and operation to perform.

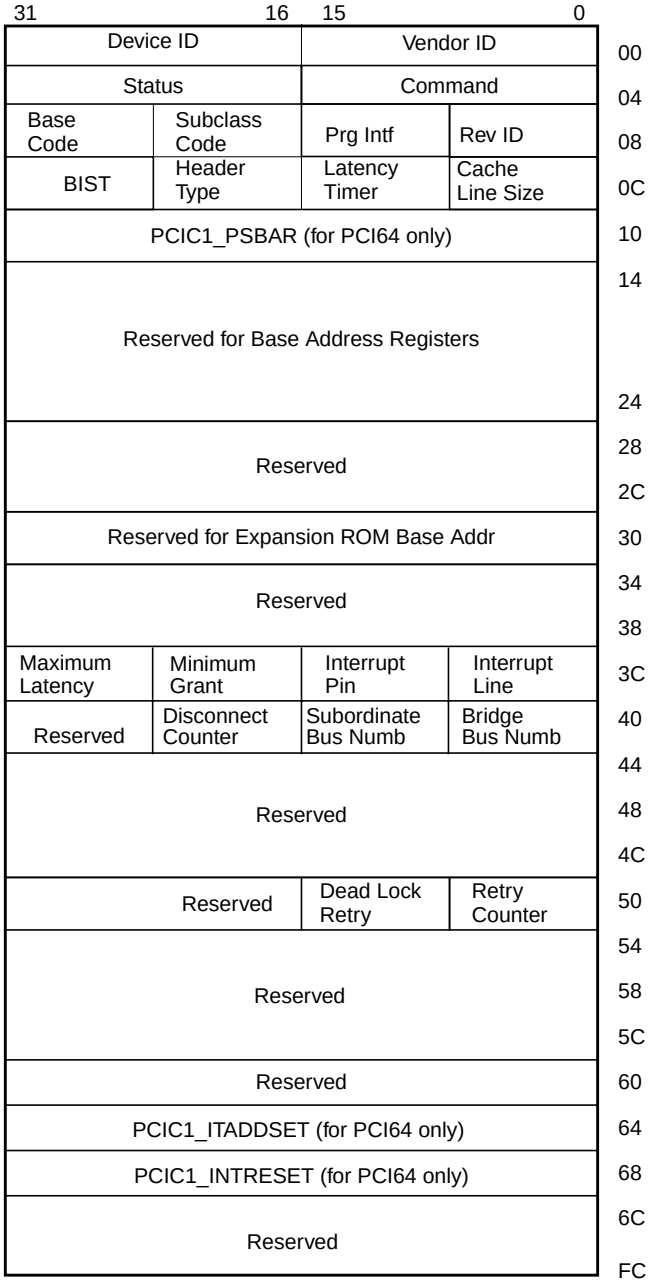


Figure 9-2. PCI Configuration Space

9.3 Standard PCI Configuration Registers

The following registers are defined as Little Endian (LE) ordering. Therefore, for software running in Big Endian (BE) mode, any access to these registers (that is not a single byte access) must utilize the

load/store byte reversal instructions when accessing these registers. Software running in LE mode can use the normal load and store instructions. There is one set of registers for the PCI 32 bit and one set for the PCI64 bit. The relative address (or register number) of these registers is specified in the PCILx_CFGADDR (Configuration Address).

Two registers PCIC1_PSBAR and PCIC1_PPBAR can be accessed and configured by the CPU or the PCI64 bus through configuration cycles.

Table 9-2. Standard PCI Configuration Registers

Relative Address	Name	Use	Page	Notes
00 to 01	PCICx_VENDID	Vendor ID Register	9-78	1
02 to 03	PCICx_DEVID	PCI Device ID Register	9-65	1
04 to 05	PCICx_CMD	Command Register	9-63	
06 to 07	PCICx_STATUS	Status Register	9-75	2
08	PCICx_REVID	Revision ID	9-74	1
09 to 0B	PCICx_CLS	PCI base class, subclass, standard programming interface	9-62	1
0C	PCICx_CACHELS	Cache Line Size	9-61	1
0D	PCICx_LATTIM	Latency Timer	9-70	
0E	PCICx_HDTYPE	Header Type	9-67	1
0F	PCICx_BIST	Built In Self Test	9-59	
10	PCIC1_PSBAR	PCI Base Address for PCI to System access (**PCI64 only**: for PCI32 see Specific PCI Host Bridge Space)	9-58	4
14	PCIC1_PPBAR	PCI Base Address for PCI to System extended access (**PCI64 only**: for PCI32 see Specific PCI Host Bridge Space)	9-57	4
3C	PCICx_INTLN	Interrupt Line	9-68	
3D	PCICx_INTPN	Interrupt Pin	9-69	
3E	PCICx_MINGNT	Minimum Grant	9-72	1
3F	PCICx_MAXLTNCY	Maximum Latency	9-71	1
40	PCICx_BUSNO	Bus Number	9-60	
41	PCICx_SUBNO	Subordinate Bus Number	9-77	
42	PCICx_DISCNT	Disconnect Counter	9-66	
50	PCICx_RETRY	Retry Counter	9-73	
51	PCIC0_ DLKRETRY	Auto Retry Counter for access in Peripheral space with potential deadlock	9-54	3
64	PCIC1_ITADDSET	Set PCI64 Inter-Processor ($\overline{\text{INT1}}$) Interrupt.	9-56	4
68	PCIC1_INTRESET	Reset of INTA,INTB,INTC,INTD on the PCI64	9-55	4

1. Read-Only Register, write is ignored
2. Writes will only reset bits in this register; write data interpreted as 1 = reset, 0 = ignore
3. Only for PCI32
4. Only for PCI64

9.4 Specific PCI Host Bridge Registers

There are two almost identical sets of registers, one for each PCI bridge that can be placed by the user in the upper 16MB of the System Memory. One CPC0_PCIBAR value (Base Address) has to be defined first for each PCI bridge; for example as shown in Figure 9-1 on page 9-2, BAR_PCI32=FF50 0000 and BAR_PCI64=FF40 0000. The register space for the PCI32 or PCI64 bridge can then be accessed by the CPU with the PCI corresponding base value loaded in the CPC0_PCIBAR (PCI Base Address Register) at address FF20 0018.

The differentiation between the PCI64 or PCI32 is made by enabling the corresponding bit in the Connectivity Configuration Register (CPC0_PCICNFR[30:31]) at address FF00 000C.

Table 9-3. Specific PCI Host Bridge Registers

Real Address	Name	Use	Page	Note
CPC0_PCIBAR + x'000F 6110	PCILx_PSEA	PCI Slave Error Address	9-106	1
CPC0_PCIBAR + x'000F 6120'	PCILx_PCIDG	PCI Diagnostic Register	9-99	
CPC0_PCIBAR + x'000F 7700'	PCILx_INTACK	Interrupt Acknowledge Cycle	9-96	1
CPC0_PCIBAR + x'000F 7800'	PCILx_PIBAR	PCI Base Address for I/O	9-100	
CPC0_PCIBAR + x'000F 7810'	PCILx_PMBAR	PCI Base Address for Memory	9-102	
CPC0_PCIBAR + x'000F 7EF0'	PCILx_CRR	Component Reset Register	9-90	
CPC0_PCIBAR + x'000F 7F20'	CPC0_PR	Personalization Register	9-104	
CPC0_PCIBAR + x'000F 7F30'	CPC0_ACR	Arbiter Control Register	9-83	
CPC0_PCIBAR + x'000F 7F40'	PCILx_MSIZE	PCI Memory Address Space Size	9-98	
CPC0_PCIBAR + x'000F 7F60'	PCILx_IOSIZE	PCI I/O Address Space Size	9-97	
CPC0_PCIBAR + x'000F 7F80'	PCILx_SMBAR	System Base Address for CPU to PCI Memory access	9-113	
CPC0_PCIBAR + x'000F 7FC0'	PCILx_SIBAR	System Base Address for CPU to PCI I/O access	9-112	
CPC0_PCIBAR + x'000F 7FD0'	PCILx_CTLRW	Configuration Register R/W	9-92	
CPC0_PCIBAR + x'000F 7FE0'		Reserved		
CPC0_PCIBAR + x'000F 8000'	PCILx_CFGADDR	PCI Configuration Address Register	9-88	2
CPC0_PCIBAR + x'000F 8010'	PCILx_CFGDATA	PCI Configuration Data Register	9-89	2

1. Read-Only Register, write is ignored
2. Little Endian registers
3. Only for PCI32
4. Only for PCI64

Table 9-3. Specific PCI Host Bridge Registers

Real Address	Name	Use	Page	Note
CPC0_PCIBAR + x'000F 8100'	PCILx_PSSIZE	PCI to System address space Size	9-109	
CPC0_PCIBAR + x'000F 8110'	PCILx_PPSIZE	PCI to System extended address space Size	9-103	
CPC0_PCIBAR + x'000F 8120'	PCILx_BARPS	System Base Address for PCI to System access	9-85	
CPC0_PCIBAR + x'000F 8130'	PCILx_BARPP	System Base Address for PCI to System extended access	9-84	
CPC0_PCIBAR + x'000F 8140'	PCIC0_PSBAR	PCI Base Address for PCI to System access (**PCI32 only**: for PCI64 see Standard PCI Configuration Space)	9-80	**3**
CPC0_PCIBAR + x'000F 8150'	PCIC0_PPBAR	PCI Base Address for PCI to System extended access (**PCI32 only**: for PCI64 see Standard PCI Configuration Space)	9-79	**3**
CPC0_PCIBAR + x'000F 8200'	PCILx_BPMDLK	Bottom of Peripheral Memory space with potential deadlock	9-87	
CPC0_PCIBAR + x'000F 8210'	PCILx_TPMDLK	Top of Peripheral Memory space with potential deadlock	9-115	
CPC0_PCIBAR + x'000F 8220'	PCILx_BIODLK	Bottom of Peripheral I/O space with potential deadlock	9-86	
CPC0_PCIBAR + x'000F 8230'	PCILx_TIODLK	Top of Peripheral I/O space with potential deadlock	9-114	
CPC0_PCIBAR + x'000F 8240'	PCILx_DLKCTRL	Deadlock Avoidance Control	9-93	
CPC0_PCIBAR + x'000F 8250'	PCILx_DLKDEV	Deadlock Device	9-95	
CPC0_PCIBAR + x'000F 8260'	PCILx_PSRCR	PCI Slave Read Control Register	9-107	
CPC0_PCIBAR + x'000F 8270'	PCILx_PSWCR	PCI Slave Write Control Register	9-110	
CPC0_PCIBAR + x'000F 8300'	PCIL1_ITADDRESET	PCI64 Reset Interrupt ($\overline{IT}$1) Addressed Register	9-82	4
CPC0_PCIBAR + x'000F 8310'	PCIL1_INTSET	Set of $\overline{G\_INTA}$, $\overline{G\_INTB}$, $\overline{G\_INTC}$, $\overline{G\_INTD}$ on PCI64	9-81	4
CPC0_PCIBAR + x'000F 9800'	PCILx_CSR	Channel Status Register	9-91	
CPC0_PCIBAR + x'000F 9810'	PCILx_PLSSR	Processor Load/Store Status Register	9-101	

1. Read-Only Register, write is ignored
2. Little Endian registers
3. Only for PCI32
4. Only for PCI64

9.5 Alphabetical List of Registers

This section lists the chip control and peripheral interface registers available in the CPC710.

CPC0_ABCNTL

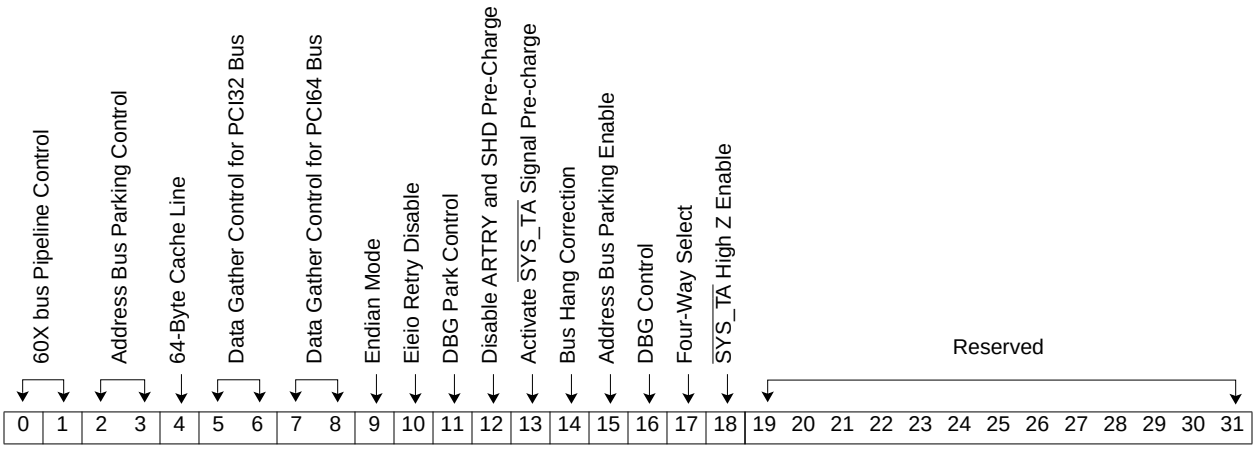
60X Arbiter Control Register

This register provides extensive control over the 60x bus arbiter operation. For a detailed description of the 60x bus arbiter, see “60x Bus Arbiter Description” on page 4-5.

Reset Value x’0000 0000’

Address x‘FF00 1030’

Access Type Read/Write


60X bus Pipeline Control
Address Bus Parking Control
64-Byte Cache Line
Data Gather Control for PCI32 Bus
Data Gather Control for PCI64 Bus
Endian Mode
Eieio Retry Disable
DBG Park Control
Disable ARTRY and SHD Pre-Charge
Activate SYS_TA Signal Pre-charge
Bus Hang Correction
Address Bus Parking Enable
DBG Control
Four-Way Select
SYS_TA High Z Enable
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**60X bus Pipeline Control** 00: Pipelining Disabled 01: One level pipelining per arbitration level is enabled (two outstanding addresses allowed) 10: Two level pipelining per arbitration level is enabled (three outstanding addresses allowed) 11: One level pipeline enabled across both arbitration levels; Selected this mode for operation with an L2 look aside controller.
2 - 3	**Address Bus Parking Control** 00: Parking Disabled 01: Parking enabled for Arbitration level 0 only 10: Parking enabled for Arbitration level 1 only 11: MRU parking enabled; Last arbitration level active is parked.
4	**64-Byte Cache Line** 0: Arbiter will grant the address bus as normal 1: Arbiter will grant a second address bus tenure to the current arbitration level, if the current arbitration level is again requesting the address bus and if first access is a burst transaction. Normal round robin grant sequence will resume after each pair of grants.
5 - 6	**Data Gather Control for CPU to PCI32 bus write** 0x: Not enabled 10: Enabled for accesses to incrementing addresses only 11: Enabled for accesses to incrementing and same addresses - NOT RECOMMENDED

Bit(s)	Description
7 - 8	**Data Gather Control for CPU to PCI64 bus write** 0x: Not enabled 10: Enabled for accesses to incrementing addresses only 11: Enabled for accesses to incrementing and same addresses - NOT RECOMMENDED
9	**Endian Mode of the PowerPC CPU** 0: 60x logic interprets data from 60x in Big Endian mode 1: 60x logic interprets data from 60x in Little Endian mode
10	**Eieio Retry Disable** 0: The CPC710 will always $\overline{\text{SYS\_ARTRY}}$ an EIEIO operation until every command in 60x queues has been dispatched to the logic units inside the CPC710. 1: The CPC710 will not $\overline{\text{SYS\_ARTRY}}$ an EIEIO operation
11	**DBG Park Control** 0: DBG signals are not parked when bus is idle 1: DBG signals are parked when bus is idle; mode to use for 0 wait state L2 look aside Bit 13 must be set to zero or this bit is ignored.
12	**Disable ARTRY and SHD Signals Pre-Charge** 0: These signals are precharged by CPC710 1: These signals are not precharged by CPC710
13	**Activate TA Signal Pre-charge** 0: These signals are precharged by CPC710 1: These signals are not precharged by CPC710
14	**CPU Bus Hang Correction** 0: No correction (same as DD2) 1: Correction of DD2 errata #11 is enabled. The time base used is the one defined for the SDRAM refresh. If a $\overline{\text{BR}}$/$\overline{\text{BG}}$ pair remains active during the time defined by the time base, then all input $\overline{\text{BR}}$ are masked during one CPU cycle such that an arbitration can be done, and thus ensure that the CPC710 can take ownership of the 60x bus and perform any pending snoop cycles.
15	**Parking Control** If MRU parking mode is enabled (bits 2-3 set to 'b11), this bit allows the 60X address bus to be parked on the last requestor for faster snoop operation. 0: CPC710 Parking disabled 1: CPC710 Parking enabled
16	**DBG Control** 0: SYS_DBG0 and SYS_DBG1 signals are driven separately 1: SYS_DBG0 and SYS_DBG1 signals are effectively the same, they are logically ORed; mode to use for L2 lookaside
17	**Quadri processor control:** 1: Arbitration of the 60X bus set for 4 CPU.
18	**$\overline{\text{SYS\_TA}}$ High Z Enable** 0: After pre-charge to up level, the SYS_TA goes to High Z. 1: After pre-charge to up level, the SYS_TA is maintained to up level. This allows the SYS_TA signal (in the case of high loading) to reach a valid (high) level in the system clock period following the pre-charge.

CPC0_ABCNTL

60X Arbiter Control Register

Bit(s)	Description
19 - 31	**Reserved**

This register contains SYS_TT, SYS_TSIZ and $\overline{\text{SYS\_TBST}}$ values that are used during a snoop transaction. These values can be changed according to the type of PowerPC processor. This register must be set if bit 25 of the CPC0_PGCHP register is programmed to 1. See “CPC0_PGCHP” on page 9-26.

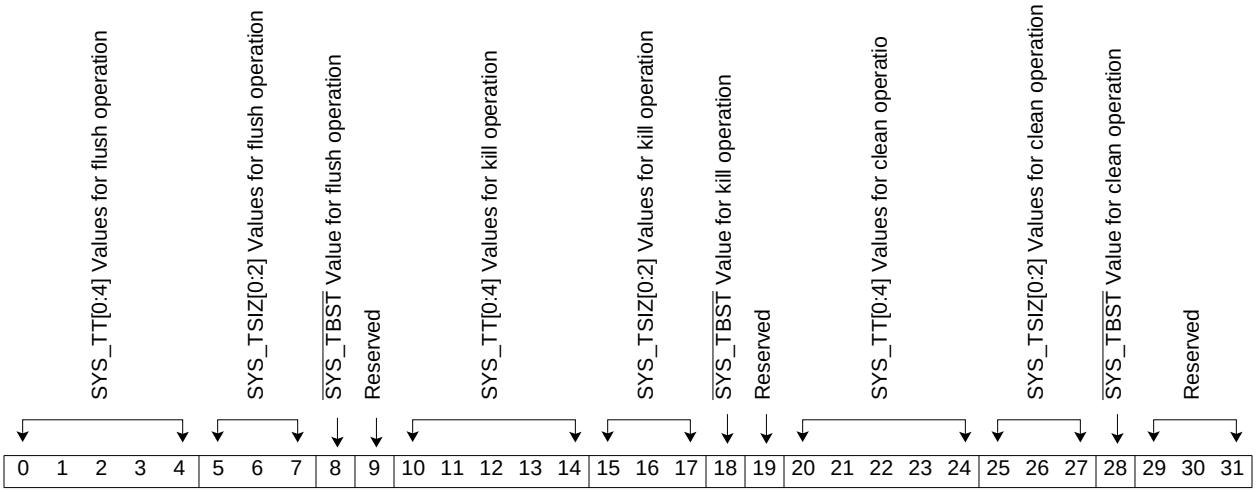
Reset Value x’0000 0000’

Address x‘FF00 1160’

Access Type Read/Write

Programming Value x’709C 2508’ b’0111 0000 1001 1100 0010 0101 0000 1000’

This setting is recommended for the PowerPC750 which is not able to perform Cache/memory coherency with Kill and Flush operation as the PowerPC 604.


SYS_TT[0:4] Values for flush operation
SYS_TSIZ[0:2] Values for flush operation
SYS_TBST Value for flush operation
Reserved
SYS_TT[0:4] Values for kill operation
SYS_TSIZ[0:2] Values for kill operation
SYS_TBST Value for kill operation
Reserved
SYS_TT[0:4] Values for clean operatio
SYS_TSIZ[0:2] Values for clean operation
SYS_TBST Value for clean operation
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 4	**SYS_TT[0:4] Values for flush operation**
5 -7	**SYS_TSIZ[0:2] Values for flush operation**
8	**$\overline{\text{SYS\_TBST}}$ Value for flush operation**
9	**Reserved**
10 - 14	**SYS_TT[0:4] Values for kill operation**
15 - 17	**SYS_TSIZ[0:2] Values for kill operation**
18	**$\overline{\text{SYS\_TBST}}$ Value for kill operation**
19	**Reserved**
20 - 24	**SYS_TT[0:4] Values for clean operation**
25 - 27	**SYS_TSIZ[0:2] Values for clean operation**
28	**$\overline{\text{SYS\_TBST}}$ Value for clean operation**
29 - 31	**Reserved**

CPC0_ATAS

Address Transfer Attribute for Snoop Register

Programming the CPC0_ATAS (Address Transfer Attribute for Snoop) Register:

When the CPC710-100 generates the following snoop cycle with TT signal on the 60x bus, the PowerPC 750 takes no action. In contrast to the PowerPC604, the PowerPC 750 does not handle cache/system memory coherency.

TT[0:4]OperationAnswer from the PowerPC750

TT[0:4]	Operation	Answer from the 750
00000	Clean Sector	No action
00100	Flush Sector	No action
01100	Kill sector	No action

To verify the coherency between Cache and System memory, with a PowerPC 750, it is necessary for the CPC710 bridge chip to modify the TT[0:4] and thus oblige the PowerPC750 to react on snoop operations with the Address only cycles on the 60x bus.

It is possible to program the CPC0_ATAS register such that the Clean, Flush, Kill code are modified in a "Snoop" code for PowerPC750

Typical changes of TT[0:4] code for the PowerPC750

Clean	TT[0:4]= 00000	-> Read	TT[0:4]	= 01010
Flush	TT[0:4]= 00100	-> RWITM	TT[0:4]	= 01110
Kill	TT[0:4]= 01100	-> RWITM	TT[0:4]	= 01110

After modification, to perform Cache/Memory coherency, the new Address only cycles are:

TT[0:4] OperationAnswer from the PowerPC750

TT[0:4]	Operation	Answer from the 750
01010	Read	Flush or Kill
01110	RWITM	Flush or Kill
01110	RWITM	Flush or Kill

Typical CPC0_ATAS programming: CPC0_ATAS[0:31]= 0x709C2508

TSIZ[0:2] and TBST can be programmed on the Address only cycles to the following recommended values TSIZ[0:2]= 000 et TBST = 1

Flush modification to RWITM:
CPC0_ATAS[0:4] <= 01110
CPC0_ATAS[5:7] <= 000
CPC0_ATAS[8] <= 1

Kill modification to RWITM
CPC0_ATAS[10:14] <= 01110
CPC0_ATAS[15:17] <= 000
CPC0_ATAS[18] <= 1

Clean modification to READ
CPC0_ATAS[20:24] <= 01010
CPC0_ATAS[25:27] <= 000
CPC0_ATAS[28] <= 1

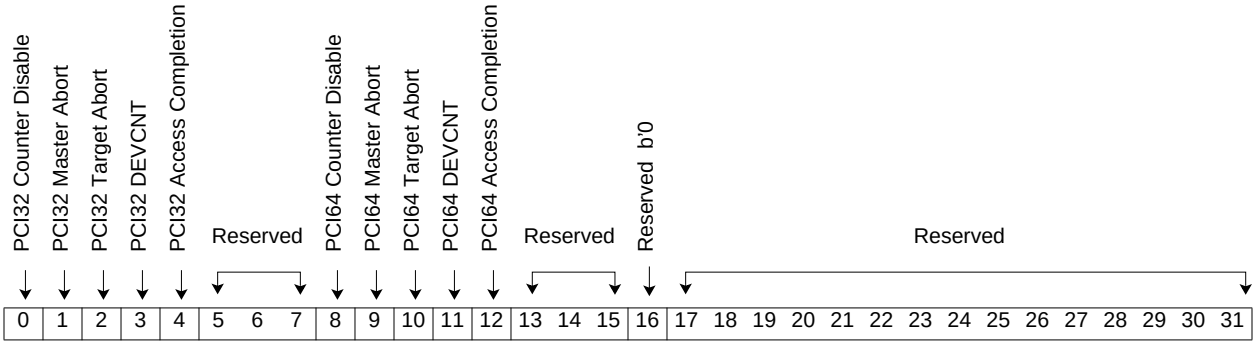
The modification is active only if bit 25 of the CPC0_PGCHP is set to 1
CPC0_PGCHP[25]= 1 - @FF001100 (Processor type 750 on).

The CPC0_AVDG control register has been introduced for verification of the fixes done on the PCI circuits in an early version of the CPC710. This register is no longer necessary in applications using the CPC710, and thus should be left to its Reset value x'0000 0000.

Reset Value x'0000 0000'

Address x'FF00 1170'

Access Type Read/Write


PCI32 Counter Disable
PCI32 Master Abort
PCI32 Target Abort
PCI32 DEVCNT
PCI32 Access Completion
Reserved
PCI64 Counter Disable
PCI64 Master Abort
PCI64 Target Abort
PCI64 DEVCNT
PCI64 Access Completion
Reserved
Reserved b'0
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**PCI32 Counter Disable** 0: TRDY, IRDY, PCICx_DISCNT Counters are not activated (The MSB is always at 1) (the PCICx_DISCNT counter is programmable - see "CPC0_AVDG" on page 9-15) 1: TRDY, IRDY, PCICx_DISCNT Counters are ACTIVATED
1	**PCI32 Master Abort** 0: Window of Master Abort is reduced to one cycle (avoid parasitic master abort detection) 1: Window of Master Abort is not reduced
2	**PCI32 Target Abort** 0: The CPC710 detects Target abort (The Frame output is taken) 1: The CPC710 never detects Target Abort but retry indefinitely accesses
3	**PCI32 DEVCNT** 0: Stop the counter down when devsel is detected 1: Do not stop the counter down
4	**PCI32 Access Completion** 0: The completion is activated when device is master and not during external exchange 1: The completion appears when the data is not the last one.
5-7	**Reserved**
8	**PCI64 Counter Enable** 0: TRDY, IRDY, PCICx_DISCNT Counters are not activated (the MSB is always at 1) PCICx_DISCNT counter is programmable - see "PCICx_DISCNT" on page 9-66 1: TRDY, IRDY, PCICx_DISCNT Counters are ACTIVATED

CPC0_AVDG

Diagnostic Register

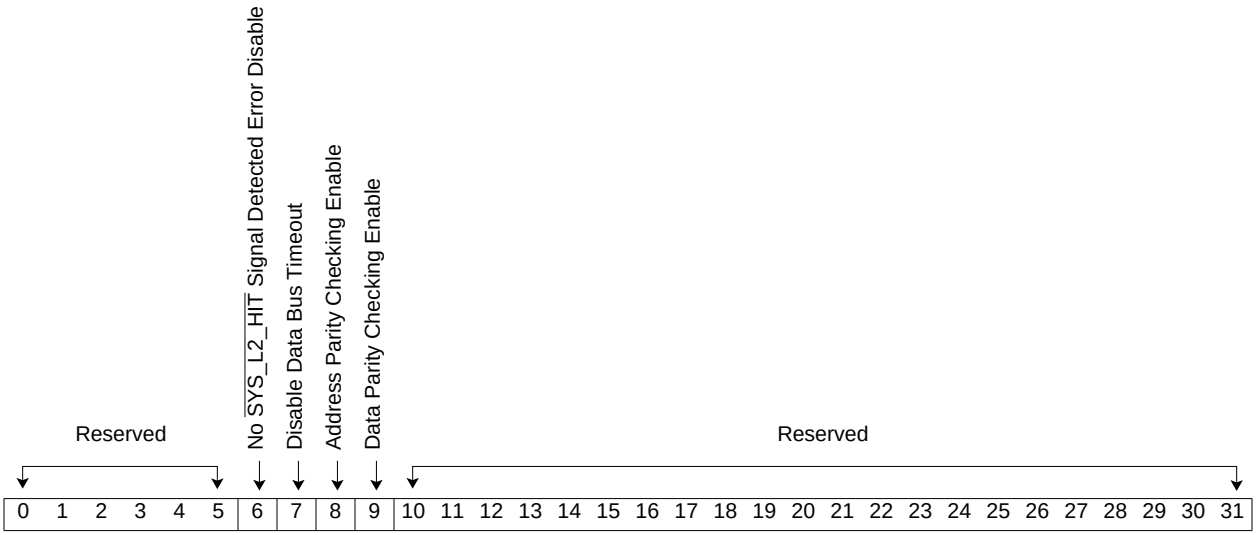
Bit(s)	Description
9	**PCI64 Master Abort** 0: Window of Master Abort is reduced to one cycle (avoid parasitic master abort detection) 1: Window of Master Abort is not reduced
10	**PCI64 Target Abort** 0: The CPC710 detects Target abort (The Frame output is taken) 1: The CPC710 never detects Target Abort but retry indefinitely accesses
11	**PCI64 DEVCNT** 0: Stop the counter down when devsel is detected 1: No stop the counter down
12	**PCI64 Access Completion** 0: The completion is activated when device is master and not during external exchange 1: The completion appears when the data is not a last.
13 - 15	**Reserved**
16	**Reserved -** Must be left to 0
17 - 31	**Reserved**

This register controls how the 60x interface logic responds when detecting an error.

Reset Value x'0000 0000'

Address x'FF00 1050'

Access Type Read/Write


Reserved
No SYS_L2_HIT Signal Detected Error Disable
Disable Data Bus Timeout
Address Parity Checking Enable
Data Parity Checking Enable
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 5	**Reserved**
6	**No SYS_L2_HIT Signal Detected Error Disable** 0: 60x logic will generate TEA on the system bus if SYS_L2_HIT signal not driven active after AACK 1: No action if SYS_L2_HIT detected inactive
7	**Disable Data Bus Timeout** In the case of timeout, the CPC710 activates the CHKSTOP and set bit 20 of the CPC0_SESR Register *(see "CPC0_SESR" on page 9-35*). 0: The CPC710 will signal error if 8ms time-out detected from DBG to TA 1: The CPC710 will not signal an error for this condition
8	**Address Parity Checking Enable** 0: 60x logic will not check address parity on the system bus 1: 60x logic will check address parity on the system bus for CPU to the CPC710 access only. In case of parity error the CHKSTOP signal is activated.
9	**Data Parity Checking Enable** 0: 60x logic will not check data parity on the system bus 1: 60x logic will check data parity on the system bus for CPU to the CPC710 access only. In case of parity error the CHKSTOP signal is activated.
10 - 28	**Reserved**

CPC0_ERRC

Error Control Register

Bit(s)	Description
29	**PCI Internal node selection for Debug** 0: PCI32 1: PCI64
30	**Internal node for Debug - Must be left to 0** 1: By multiplexing CPC710 Internal nodes are on the I/O pins SYS_ADDR32 SYS_ADDR33 SYS_ADDR34 SYS_ADDR35 SYS_ADDRP4
31	**Internal node for Debug - Must be left to 0** 1: By multiplexing CPC710 Internal nodes are on the I/O pins: $\overline{\text{SYS\_SRESET2}}$ $\overline{\text{SYS\_SRESET3}}$ $\overline{\text{SYS\_MCP2}}$ $\overline{\text{SYS\_MCP3}}$

CPC0_GPDIR

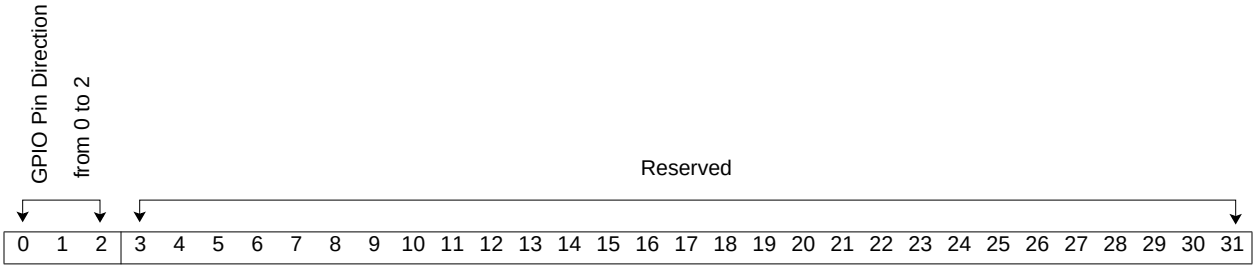
GPIO Direction Register

This register sets the direction of signals (input or output) on pins GPIO0, GPIO1 and GPIO2.

Reset Value x'0000 0000

Address x'FF00 1130'

Access Type Read/Write



Bit(s)	Description0
0	**GPIO 0 Pin Direction** 0: Input 1: Output
1	**GPIO 1 Pin Direction** 0: Input 1: Output
2	**GPIO 2 Pin Direction** 0: Input 1: Output
3 - 31	**Reserved**

CPC0_GPIN

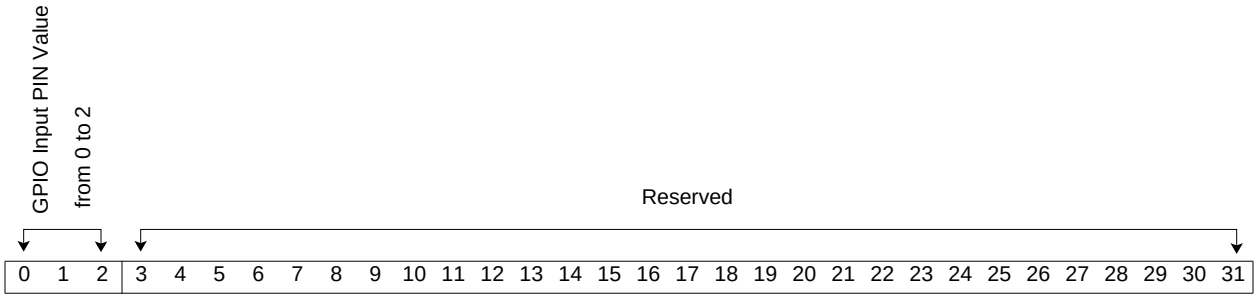
GPIO Input Value Register

This register stores values of the signal on pins GPIO0, GPIO1, GPIO2 if it is defined as input.

Reset Value x'0000 0000

Address x'FF00 1140'

Access Type Read Only



Bit(s)	Description
0	**GPIOO Input Pin Value**
1	**GPIO1 Input Pin Value**
2	**GPIO2 Input Pin Value**
3 - 31	**Reserved**

CPC0_GPOUT

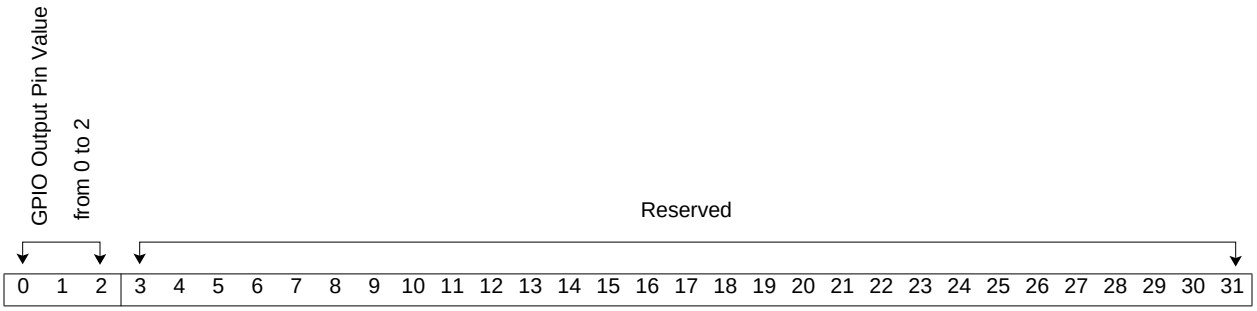
GPIO Output Value Register

This register stores values of signal on pins GPIO0, GPIO1, GPIO2 if defined as output.

Reset Value x'0000 0000

Address x'FF00 1150'

Access Type Read/Write



Bit(s)	Description
0	**GPIOO Output Pin Value**
1	**GPIO1 Output Pin Value**
2	**GPIO2 Output Pin Value**
3 - 31	**Reserved**

CPC0_MPSR

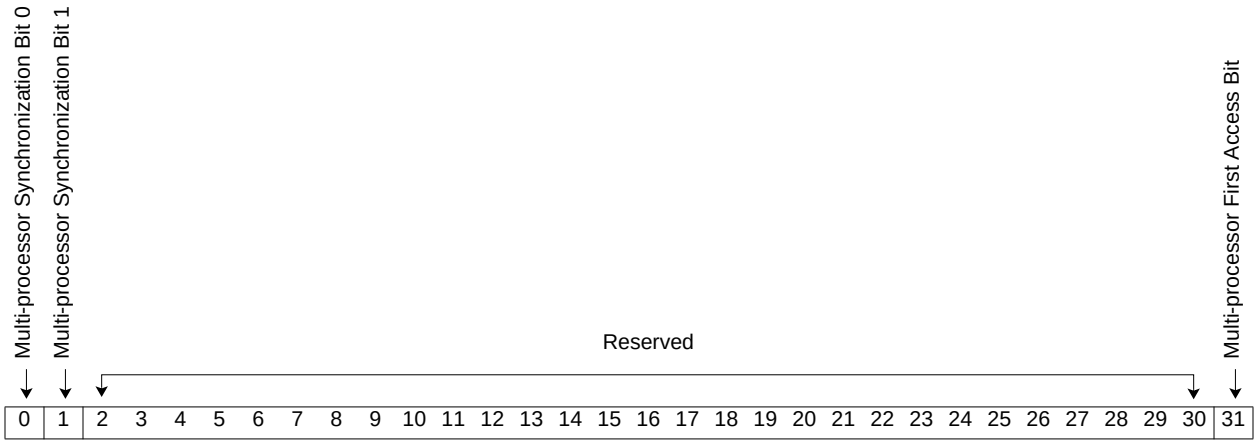
Multiprocessor Semaphore

This register is used by the IPL boot code to facilitate bring-up of processors in an MP environment. It provides a first access bit, BIT 31, that allows a method for processors to determine which processor is the master, since both processors are active after power on. BIT 31 contains a value of 0 after power on reset. When the first processor read occurs to this register, BIT 31 returns a value of 0. All subsequent reads of this register return a value of 1 for BIT 31. In addition to the First Access Bit, bits 0 and 1 provide semaphores for use by the firmware during boot time and are utilized until system memory has been initialized and tested.

Reset Value x'0000 0000'

Address x'FF00 1010'

Access Type Read/Write


Multi-processor Synchronization Bit 0
Multi-processor Synchronization Bit 1
Reserved
Multi-processor First Access Bit
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Multi-processor Synchronization Bit 0.** Used for communication between processors at IPL time
1	**Multi-processor Synchronization Bit 1.** used for communication between processors at IPL time
2 - 30	**Reserved**
31	**Multi-processor First Access Bit**. Read Only; Set after read 0: Initial power on value; Indicates first read of this register. 1: Indicates that this register has been read at least once previously.

CPC0_PCIBAR

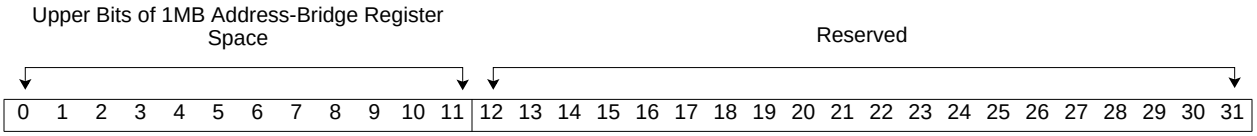
PCI Base Address Register

This register is written by software to indicate to the PCI bridge where its register space is located in the 4 GB system addressing space. There are no restrictions placed on the value of this register other than it must not overlap other extents defined for the system.

Reset Value x'0000 0000

Address x'FF20 0018'

Access Type Read/Write



Bit(s)	Description
0 - 11	**Upper Bits of 1 MB Address for Bridge Register Space**
12 - 31	**Reserved.** (Assumed to be X'0 0000')
Note: The start address is assumed to be on a 1 MB boundary.	

CPC0_PCICNFR

Connectivity Configuration Register

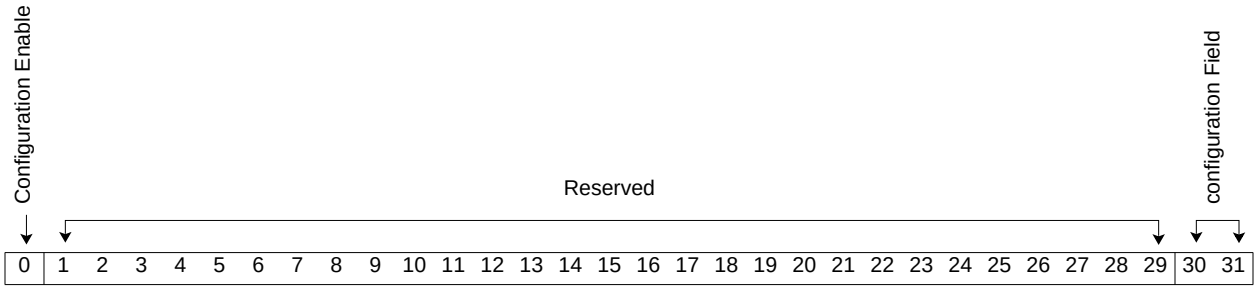
CPC0_PCICNFR supports the initialization and configuration of the PCI bridge facilities (CPC0_PCIBAR register) . This register provides the unique setup signal required to insure that only one device will respond to configuration addresses at a time. Software must adhere to the following restrictions for configuration:

- A write to the CPC0_PCICNFR register must be followed by a SYNC operation or a read of the register.

Reset Value x'0000 0000'

Address x'FF00 000C'

Access Type Read/Write



Bit(s)	Description
0	**Configuration Enable** 0: Disable Configuration access 1: Enable Configuration access defined as described in bits 30 - 31
1 - 29	**Reserved**
30 - 31	**Configuration Field** 0x: No action 10: Configuration access directed to PCI32 bus 11: Configuration access directed to PCI64 bus

CPC0_PCIENB

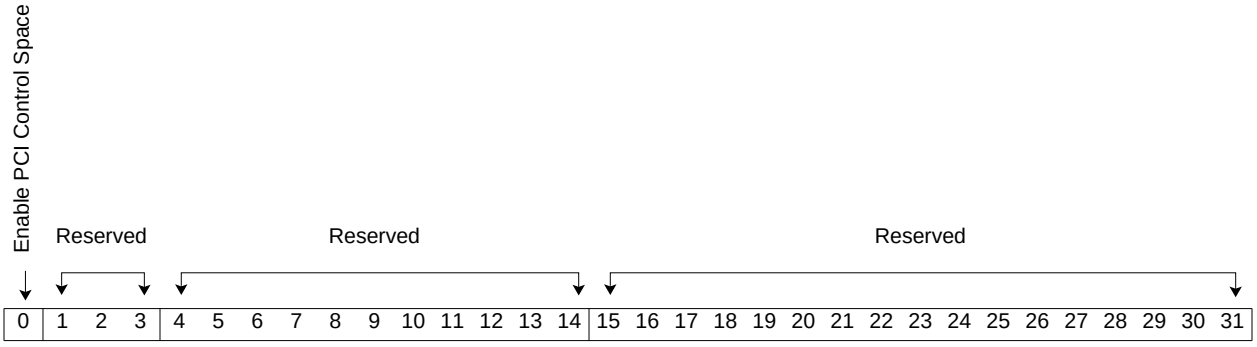
PCI Base Address Register Enable Register

This register provides a mechanism for software to disable the PCI bridge logic from decoding the address space pointed to by CPC0_PCIBAR. This register is primarily used at power on time when the CPC0_PCIBAR has not been initialized.

Reset Value x'0000 0000'

Address x'FF20 1000'

Access Type Read/Write



Bit(s)	Description
0	**Enable PCI Control Space** 0: PCI Bridge only responds to configuration cycles. The other access to CPC0_PCIBAR are Inhibited: Normal mode 1: PCI Bridge responds to address space specified in the CPC0_PCIBAR register (PCI configuration phase)
1 - 3	**Reserved**
4 - 31	**Reserved**

CPC0_PGCHP

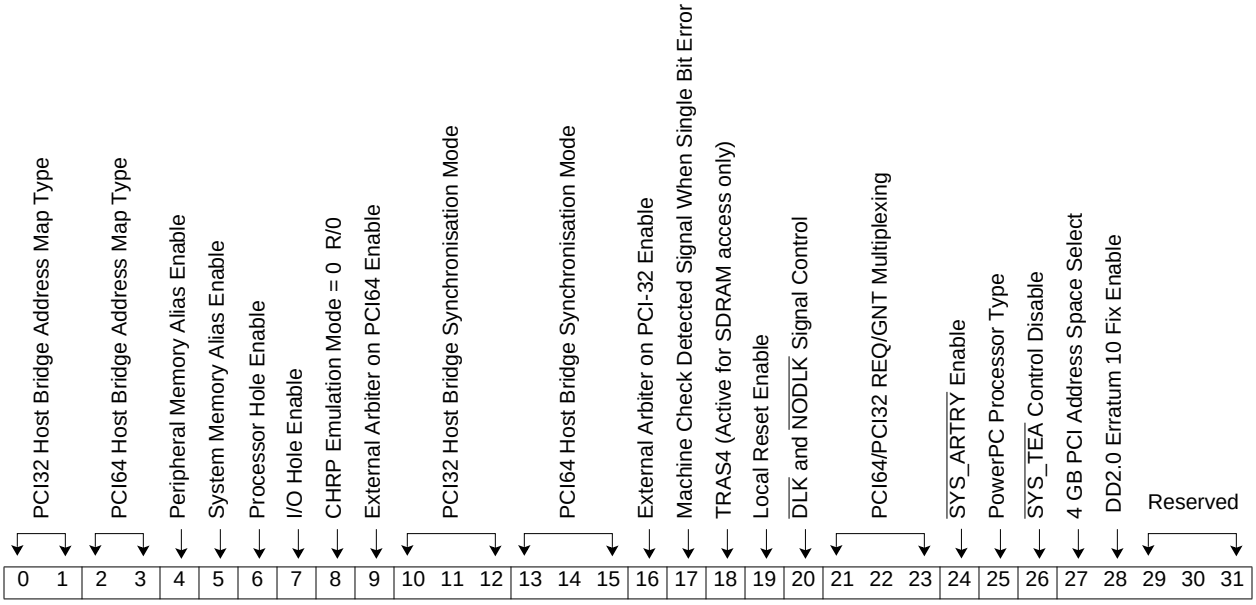
Chip Programmability Register

The bits in this register are used to disable certain errata fixes and also allow for selection of additional functions such as the PCI mapping in Flexible, PREP, or CHRP mode.

Reset Value x'0000 0000'

Address x'FF00 1100'

Access Type Read/Write


PCI32 Host Bridge Address Map Type
PCI64 Host Bridge Address Map Type
Peripheral Memory Alias Enable
System Memory Alias Enable
Processor Hole Enable
I/O Hole Enable
CHRP Emulation Mode = 0 R/0
External Arbiter on PCI64 Enable
PCI32 Host Bridge Synchronisation Mode
PCI64 Host Bridge Synchronisation Mode
External Arbiter on PCI-32 Enable
Machine Check Detected Signal When Single Bit Error
TRAS4 (Active for SDRAM access only)
Local Reset Enable
DLK and NODLK Signal Control
PCI64/PCI32 REQ/GNT Multiplexing
SYS_ARTRY Enable
PowerPC Processor Type
SYS_TEA Control Disable
4 GB PCI Address Space Select
DD2.0 Erratum 10 Fix Enable
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0-1	**PCI32 Host Bridge Address Map Type.** Controls which address map is used. See "Address Maps" on page 2-1. 00: PREP mode 01: CHRP mode with the PCI32 Host Bridge defined as the PCI Host Bridge 0 (PHB0) 10: FPHB mode (Base address on PCI32 is used) 11: CHRP mode with the PCI-32 Host Bridge defined as the PCI Host Bridge 1 (PHB1) - The PCI64 is PHB0
2-3	**PCI64 Host Bridge Address Map Type.** Controls which address map is used. See "Address Maps" on page 2-1 00: PREP mode 01: CHRP mode with the PCI64 Host Bridge defined as the PCI Host Bridge 0 (PHB0) 10: FPHB mode (Base address on PCI64 is used) 11: CHRP mode with the PCI64 Host Bridge defined as the PCI Host Bridge 1 (PHB1) - The PCI32 is PHB0

Bit(s)	Description
4	**Peripheral Memory Alias Enable** This bit is used in **CHRP mode** only for processor-initiated transactions to PCI Memory 0: No **translation** for processor access in the Peripheral-memory-alias space 1: Translate an address in the Peripheral-memory-alias space (BIM to TPM0; 16 MB range) so that this address range becomes 0 to (16 MB -1) in the PCI Memory Space.
5	**System Memory Alias Enable** This bit is used in **CHRP mode** only for PCI-initiated transactions to System Memory 0: No **reponse** for PCI access in the System-memory-alias space 1: Translate an address in the System-memory-alias space (BIM to TPM0; 16 MB range) so that this address range becomes 0 to (16 MB -1) in the System Memory Space.
6	**Processor-Hole Enable** This bit is used in **CHRP mode** only for Processor-initiated transactions 0: Forwards processor-initiated transactions in the range 640 KB to (768KB -1) to the System Memory Space 1: Forwards processor-initiated transactions in the range 640 KB to (768KB -1) to the PCI Memory Space
7	**IO-Hole Enable** This bit is used in **CHRP mode** only for PCI-initiated transactions 0: Untranslate an address in the range 640 KB to (1 MB -1) and then send to System Memory Space 1: No **response** to PCI access in range 640 KB to (1 MB -1)
8	**Emulation Mode** This bit is **Read Only** and indicates that the CPC710 doesn't support the optional emulation mode of the CHRP mode 0: CHRP Emulation Mode not supported
9	**External Arbiter on PCI64 Enable.** Read only status bit. 0: Internal arbiter is activated 1: Internal arbiter is deactivated
10 - 12	**PCI32 Host Bridge Synchronisation Mode** 001: Fast synchronisation xx0: Double synchronisation
13 - 15	**PCI64 Host Bridge Synchronisation Mode** 001: Fast synchronisation xx0: Double synchronisation
16	**External Arbiter on PCI32 Enable.** Read only status bit. 0: Internal arbiter is activated 1: Internal arbiter is deactivated
17	**Machine Check Detected Signal When ECC Single Bit Error** 0: $\overline{\text{SYS\_MCP}}$ signal not driven 1: $\overline{\text{SYS\_MCP}}$ signal is driven only if CPC0_PGCHP[26] =1 ($\overline{\text{SYS\_TEA}}$ Control Disable)
18	**TRAS4 (Active for SDRAM access only)** 0: t_{RASmin} = 5 * clock 1: t_{RASmin} = 4 * clock

CPC0_PGCHP

Chip Programmability Register

Bit(s)	Description
19	**Local Reset Enable** 0: PCI64 Signal G_RESETOUT is not driven 1: If external arbiter on PCI64 bus then G_RESETOUT is driven.
20	**DLK and NODLK Signal Control (must be left as 0)** 0: DLK and NODLK are enabled.
21	**PCI 64 REQ-GNT 5 signal multiplexing with PCI 32 REQ-GNT 4 control** 1: G_REQ_5 becomes P_REQ_4 G_GNT_5 becomes P_GNT_4
22	**PCI 64 REQ-GNT 6 signal multiplexing with PCI 32 REQ-GNT 5** 1: G_REQ_6 becomes P_REQ_5 G_GNT_6 becomes P_GNT_5
23	**PCI 64 REQ-GNT 7 signal multiplexing with PCI 32 REQ-GNT 6** 1: G_REQ_7 becomes P_REQ_6 G_GNT_7 becomes P_GNT_6
24	**Auto Retry Enable (must be left as 0 if internal Deadlock avoidance circuit is enabled)** 0: SYS_ARTRY is not always driven 1: SYS_ARTRY is always driven when the access is in Peripheral Memory or I/O space with potential deadlock and CPC710 input NODLK = 1. This setting permits use of external circuit control with DLK and NODLK
25	**PowerPC Processor Type** (see “CPC0_ATAS” on page 9-13). 0: 604 1: 750 or later version
26	**TEA Control Disable** 0: SYS_TEA is driven 1: SYS_TEA is not driven but Machine Check signal is only if CPC0_PGCHP[17] = 1.
27	**4 GB memory adress space for PCI access** 0: The maximum memory space size is 2 GB 1: The maximum memory space size is 4 GB (available for FPHB Mode or CHRP mode only)
28	**DD2.0 ERRATA#10 correction for fast back-to-back mode** 0: ERRATUM is not corrected 1: ERRATUM is corrected
29 - 31	**Reserved** R/W - Must be left to 0

CPC0_PIDR

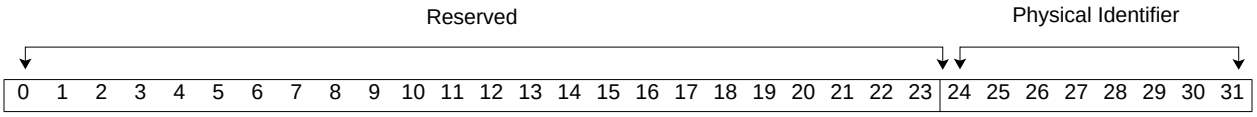
Physical Identifier Register

This register provides a unique number for each processor (or any 60x bus master) reading this location. It is primarily used by processors to differentiate themselves in multiprocessor configurations. When this register is read, the CPC710 latches the current processor's $\overline{\text{SYS\_BR}}$/$\overline{\text{SYS\_BG}}$ pair into this register which physically identifies the processor. Each processor has a unique $\overline{\text{SYS\_BR}}$/$\overline{\text{SYS\_BG}}$ pair connected to it.

Reset Value x'0000 0000'

Address x'FF00 0008'

Access Type Read Only



Bit(s)	Description
0 - 23	**Reserved**
24 - 31	**Physical Identifier.** The CPC710 responds with two values for this field: x'00': Indicates processor associated with $\overline{\text{BR}}$0 and $\overline{\text{BG}}$0 pins. x'01': Indicates processor associated with $\overline{\text{BR}}$1 and $\overline{\text{BG}}$1 pins x'02': Indicates processor associated with $\overline{\text{BR2}}$ and $\overline{\text{BG2}}$ pins. x'03': Indicates processor associated with $\overline{\text{BR3}}$ and $\overline{\text{BG3}}$ pins

CPC0_RGBAN0

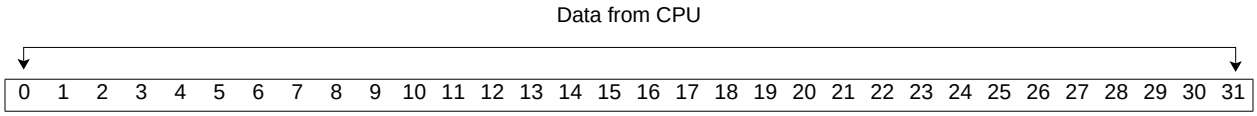
Free Register 0

This register contains data coming from the CPU.

Reset Value x'0000 0000'

Address x'FF00 1110'

Access Type Read/Write



Bit(s)	Description
0 - 31	**Data from CPU**

CPC0_RGBAN1

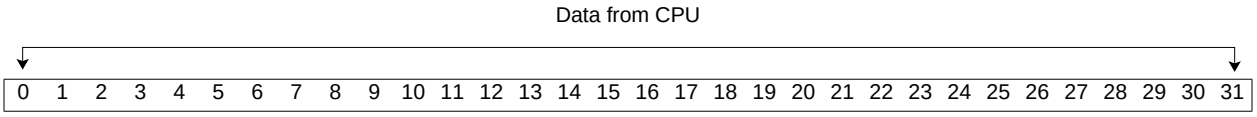
Free Register 1

This register contains data coming from the CPU.

Reset Value x'0000 0000

Address x'FF00 1120'

Access Type Read/Write



Bit(s)	Description
0 - 31	**Data from CPU**

CPC0_RSTR

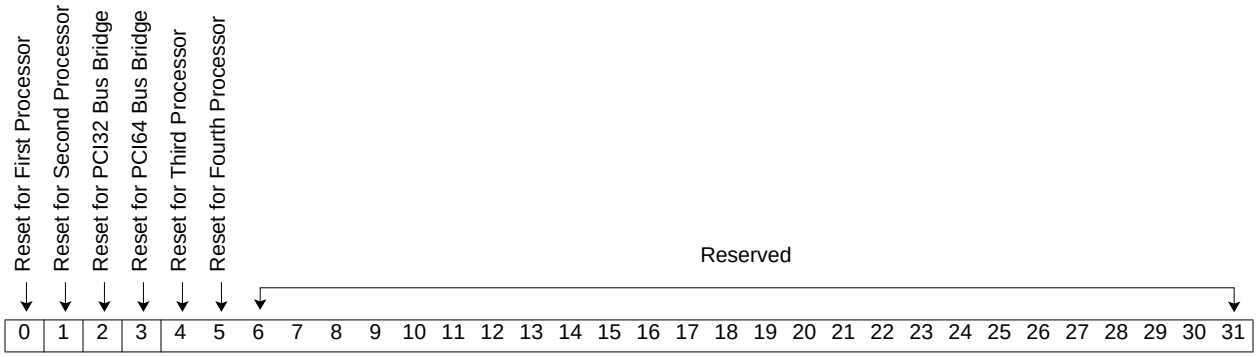
Connectivity Reset Register

This register provides a means to individually reset devices on the 60x bus.
Bits 0, 1, 4 and 5 directly control $\overline{\text{SYS_HRESET}}$0, $\overline{\text{SYS_HRESET}}$1, $\overline{\text{SYS_HRESET}}$2, $\overline{\text{SYS_HRESET}}$3 respectively. The remaining two bits 2 and 3 control reset signals that are internal to the CPC710.

Reset Value x'CC00 0000'

Address x'FF00 0010'

Access Type Read/Write


Reset for First Processor
Reset for Second Processor
Reset for PCI32 Bus Bridge
Reset for PCI64 Bus Bridge
Reset for Third Processor
Reset for Fourth Processor
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Reset For First Processor** 0: $\overline{\text{SYS_HRESET}}$0 signal is active 1: $\overline{\text{SYS_HRESET}}$0 signal is inactive
1	**Reset For Second Processor** 0: $\overline{\text{SYS_HRESET}}$1 signal is active 1: $\overline{\text{SYS_HRESET}}$1 signal is inactive
2	**Reset of the internal PCI 32 Bus Bridge of the CPC710** 0: Reset active 1: Reset inactive
3	**Reset of the internal PCI64 Bus Bridge of the CPC710** 0: Reset active 1: Reset inactive
4	**Reset For Third Processor** 0: $\overline{\text{SYS_HRESET}}$2 signal is active 1: $\overline{\text{SYS_HRESET}}$2 signal is inactive
5	**Reset For Fourth Processor** 0: $\overline{\text{SYS_HRESET}}$3 signal is active 1: $\overline{\text{SYS_HRESET}}$3 signal is inactive
6-31	**Reserved**

This register permit to control timers clocked from the PCI32 clock.

Reset Value x'8022 4470'

Address x'FF00 0020'

Access Type Read/Write

Bit(s)	Description
0-9	**Time Base for DRAM refresh**
10-15	**Time Base for Soft Reset (Controled by CPC0_SRST Register)**
16-19	**Time Base for Software Power On Reset (Controled by CPC0_SPOR Register)**
20-23	**Time Base for Bus Timeout**
24-27	**Time Base for Sdram Initialisation Phase**
28 - 31	**Reserved**

Programming example :

The times shown below are obtained with the PCI 32 clock at 33MHz with the register CPC0_RTBR[0:31] set at : 32'h80224470

With other PCI32 frequency operation, setting should be made such that the

times are matching closely with these values.

Bits	Type of timing	Value	Clocking	Period	Time
0:9	SDRAM refresh time	10.0000.0000	33MHz PCI32 clock	512 x 30 ns	15.360 us
10:15	Soft reset time	100010	33MHz PCI32 clock	34 x 30 ns	1.020 us
16:19	Software power on reset time	0100	33MHz PCI32 clock	4 x2**16 x 30 ns	7.86432 ms
20:23	Bus Timeout	0100	33MHz PCI32 clock	4 x2**16 x 30 ns	7.86432 ms
24:27	SDRAM init phase counter	0111	33MHz PCI32 clock	7 x 1024 x 30 ns	215 us.

CPC0_SEAR

System Error Address Register

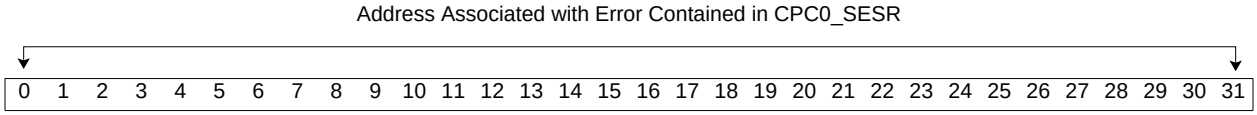
This register contains the CPU address associated with the error that is logged in the SESR register described previously. This register is only updated for errors that are due to CPU initiated transfers. The address for errors that result from transfers initiated by PCI masters or DMA controller are located in error registers contained in the PCI bridge logic or the DMA controller logic.

In the case of dual-processor implementation, this register will contain only the address of the first error detected.

Reset Value x'0000 0000'

Address x'FF00 1070'

Access Type Read/Write



Bit(s)	Description
0 - 31	**Address Associated with Error Contained in CPC0_SESR**

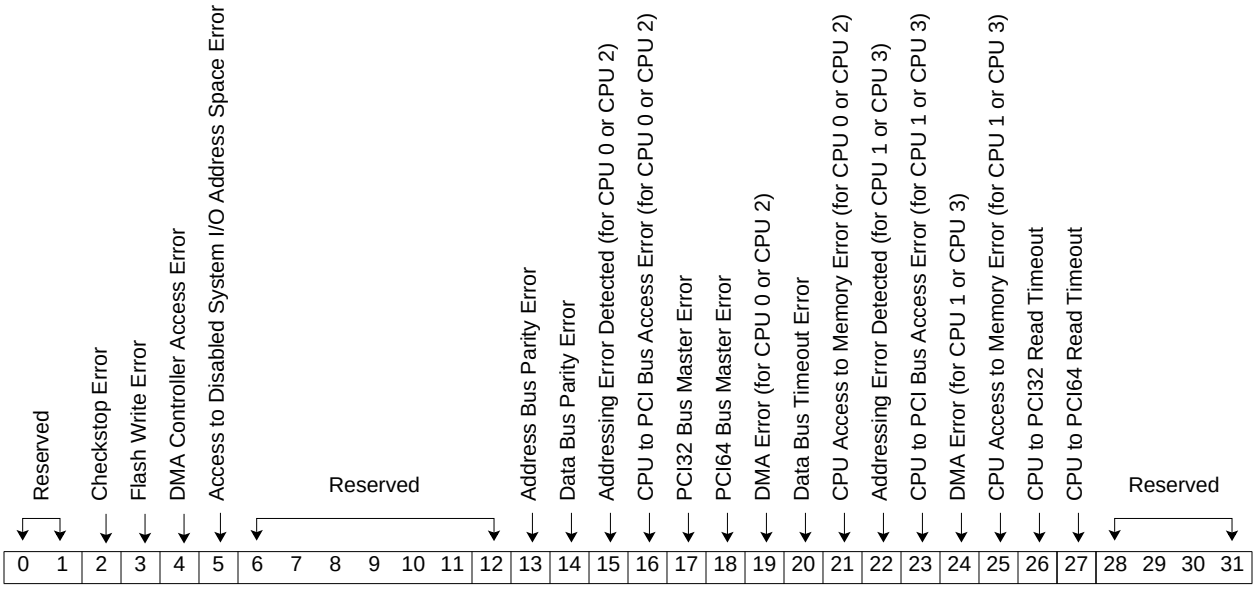
This register is the primary error status register for the CPC710 and should be read first after a Machine Check interrupt occurs ($\overline{MCP}$0, $\overline{MCP}$1,$\overline{MCP}$2 or $\overline{MCP}$3 activated by the CPC710) . All errors that result from CPU initiated transfers are logged in this register. Errors resulting from transfers initiated by a PCI Master or by the DMA controller will result in bits 17, 18, or 19 being set and require software to interrogate additional error registers in the PCI bridge logic and the DMA controller logic. Bit 16, CPU to PCI Bus error, will also require software to interrogate additional error registers in the PCI bridge logic.

The bits 22, 23, 24, 25 that are available for read after a CPU1 Machine Check interrupt have the same meaning as errors reported on bits 15, 16, 19, 21 for CPU0.
Software is responsible for writing zeros to this register in order to clear or deactivate the appropriate SYS_MCP0:3 signal.

Reset Value x'0000 0000'

Address x'FF00 1060'

Access Type Read/Write


Reserved
Checkstop Error
Flash Write Error
DMA Controller Access Error
Access to Disabled System I/O Address Space Error
Reserved
Address Bus Parity Error
Data Bus Parity Error
Addressing Error Detected (for CPU 0 or CPU 2)
CPU to PCI Bus Access Error (for CPU 0 or CPU 2)
PCI32 Bus Master Error
PCI64 Bus Master Error
DMA Error (for CPU 0 or CPU 2)
Data Bus Timeout Error
CPU Access to Memory Error (for CPU 0 or CPU 2)
Addressing Error Detected (for CPU 1 or CPU 3)
CPU to PCI Bus Access Error (for CPU 1 or CPU 3)
DMA Error (for CPU 1 or CPU 3)
CPU Access to Memory Error (for CPU 1 or CPU 3)
CPU to PCI32 Read Timeout
CPU to PCI64 Read Timeout
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**Reserved**
2	**Checkstop Error** (0=No Error) 1: CPC710 initiated checkstop occurred
3	**Flash Write Error** (0=No Error) 1: Write to flash occurred when not enabled
4	**DMA Controller Access Error** (0=No Error) 1: Access performed to DMA Controller when not enabled (see *"DMA0_GSCRP, DMA0_GSCRU" on page 9-45)*
5	**Access to Disabled System I/O Address Space Error** (0=No Error) 1: Access performed to System I/O address space that is not enabled

CPC0_SESR

System Error Status Register

Bit(s)	Description
6 - 12	**Reserved**
13	**Address Bus Parity Error** (0=No Error) 1: 60x bus address parity error detected by the CPC710
14	**Data Bus Parity Error** (0=No Error) 1: 60x bus data parity error detected by the CPC710
15	**Addressing Error Detected (for CPU 0 or CPU 2)** (0=No Error) 1: Addressing error This bit is set when the following conditions are true : - CPC710 is not selected by the current CPU access - SYS_L2_HIT signal is not driven active after $\overline{\text{AACK}}$ by any agent on the system bus - CPC0_ERRC[6] is set to 0 (See “CPC0_ERRC” on page 9-17.) The CPC710 will signal an addressing error by generating TEA or MCP on the system bus (dependent on programming of CPC0_PGCHP[26]. See “CPC0_PGCHP” on page 9-26.)
16	**CPU to PCI Bus Access Error (for CPU 0 or CPU 2)** (0=No Error) 1: Error occurred on PCI32 or PCI64 bus while servicing processor load/store request
17	**PCI32 Bus Master Error** (0=No Error) 1: Error occurred during PCI master initiated operation
18	**PCI64 Bus Master Error** (0=No Error) 1: Error occurred during PCI master initiated operation
19	**DMA Error (for CPU 0 or CPU 2)** (0=No Error) 1: Error occurred during DMA transfer
20	**Data Bus Timeout Error** (0=No Error) 1: Indicates that the CPC710 has detected a 8ms time-out between DBG to last $\overline{\text{SYS\_TA}}$ or $\overline{\text{SYS\_TEA}}$. In this case of error the CPC710 activates also the $\overline{\text{CHKSTOP}}$ signal.
21	**CPU Access to Memory Error (for CPU 0 or CPU 2)** (0=No Error) 1: Error occurred during an access by the CPU to memory; Error logged in SDRAM0_MESR and SDRAM0_MEAR
22	**Addressing Error Detected (FOR CPU 1 or CPU 3)** (0=No Error) 1: Addressing error. This bit is set when the following conditions are true : - CPC710 is not selected by the current CPU access - SYS_L2_HIT signal is not driven active after $\overline{\text{AACK}}$ by any agent on the system bus - CPC0_ERRC[6] is set to 0 (See “CPC0_ERRC” on page 9-17.) The CPC710 will signal an addressing error by generating TEA or MCP on the system bus (dependent on programming of CPC0_PGCHP[26]. See “CPC0_PGCHP” on page 9-26.)
23	**CPU to PCI Bus Access Error (for CPU 1 or CPU 3)** (0=No Error) 1: Error occurred on PCI32 or PCI64 bus while servicing processor load/store request
24	**DMA Error (for CPU 1 or CPU 3)** (0=No Error) 1: Error occurred during DMA transfer
25	**CPU Access to Memory Error (for CPU 1 or CPU 3)** (0=No Error) 1: Error occurred during an access by the CPU to memory;

Bit(s)	Description
26	**CPU to PCI32 Read Timeout** (0=No Error) 1: Error occurred during an read access in a deadlock area
27	**CPU to PCI64 Read Timeout** (0=No Error) 1: Error occurred during an read access in a deadlock area
28-31	**Reserved**

CPC0_SIOC0

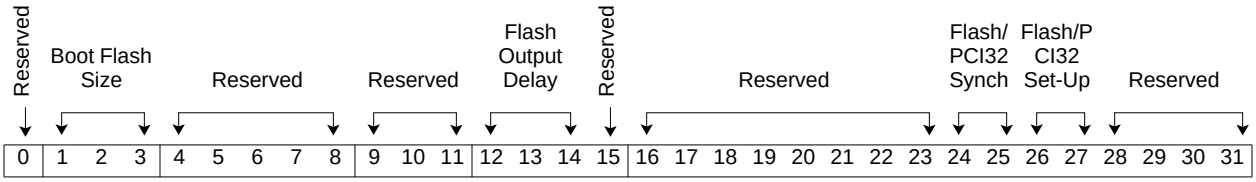
System I/O Control

This register provides initialization and control of the Boot FLASH and the Extended FLASH devices to which the CPC710 interfaces.

Reset Value x'0000 0000'

Address x'FF00 1020'

Access Type Read/Write


Reserved
Boot Flash Size
Reserved
Reserved
Flash Output Delay
Reserved
Reserved
Flash/ PCI32 Synch
Flash/P CI32 Set-Up
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Flash Timing Control** 0: The CPC710 does not use values programmed in FCNT, LATMAX and LADRMAX fields (see below) but uses values depending of the PLL_RANGE1 input. 1: The CPC710 use values programmed in FCNT, LATMAX and LADRMAX fields (see below)
1 - 3	**Boot Flash Size** 000: 2.0 MB - The CPC710 initiates FLASH access for addresses x'FFE0 0000' to x'FFFF FFFF' 001: 1.0 MB - The CPC710 initiates FLASH access for addresses x'FFF0 0000' to x'FFFF FFFF' 011: 0.5 MB - The CPC710 initiates FLASH access for addresses x'FFF0 0000' to x'FFF7 FFFF' 111: Reserved
4 - 8	**Reserved**
9 - 11	**Reserved**. These bit should be left to zero.
12 - 14	**FCNT: Flash output delay (no action if bit 0 is left to 0)** Number of system Clock cycles from signal XADR_LAT falling edge to signal FLASH_OE_ (or FLASH_WE_ if write) rising Edge. 000: 12 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 0) 001: 13 Clock cycles 010: 14 Clock cycles 011: 15 Clock cycles 100: 16 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 1) Others: 12 Clock cycles
15	**Reserved**
16 - 23	**Reserved**

Bit(s)	Description
24 - 25	**LATMAX (no action if bit 0 is left to 0)** Number of system Clock cycles needed for the CPC710 from driving FLASH address on the PCI32 bus to the resynchronisation with the PCI32 Clock. 00: 5 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 0) 01: 6 Clock cycles 10 : 7 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 1) 11: 4 Clock cycles
26 - 27	**LADRMAX (no action if bit 0 is left to 0)** Number of system Clock cycles for activation of the signal XADR_LAT from the time where the CPC710 drive the Flash address on the PCI32 bus. 00: 3 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 0) 01: 4 Clock cycles 10: 5 Clock cycles (value used if bit 0 is left to 0 and PLL_RANGE1 = 1) 11: 2 Clock cycles
28 - 31	**Reserved**

CPC0_SIOC1

System I/O Control 1

This register provides initialization and control of the Extended FLASH devices attached to the CPC710 through the PCI32 interface.

Reset Value x'0000 0000'

Address x'FF00 1090'

Access Type Read/Write

Bit(s)	Description
0	**Flash Priority** 1: the priority access of the Boot ROM or the Extended Flash versus the PCI agents is increased.
1-3	**Flash Size** 000: No Optional Flash Space 001: 16 MB 010: 32 MB 011: 64 MB 100: 128 MB 101: 256 MB Other:No Optional Flash Space
4-11	**System Base Address** This registers contains the upper bits of the System Base address of where FLASH is mapped to. The boundary alignment for the FLASH Space must be an integer multiple of the size of the space.
12-13	**FLASH DATA BUS SIZE** 00: 8 bit system FLASH space 01: 16 bit system FLASH space 10: 32 bit system FLASH space 11: Reserved
14-31	**Reserved** - Read Only

CPC0_SPOR

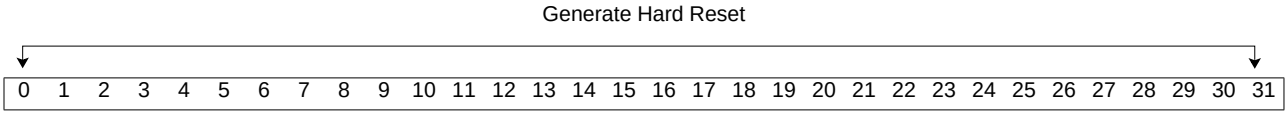
Software Power-On Reset Control Register

This register provides a mechanism for software to initiate a hard reset to the system.
The CPC710 will activate resets to all processors (SYS_HRESET0 to 3) and PCI devices.

Reset Value x'0000 0000'

Address x'FF00 00E8'

Access Type Write Only



Bit(s)	Definition
0 - 31	**Generate Hard Reset** A write to this register will initiate a power on reset.

CPC0_SRST

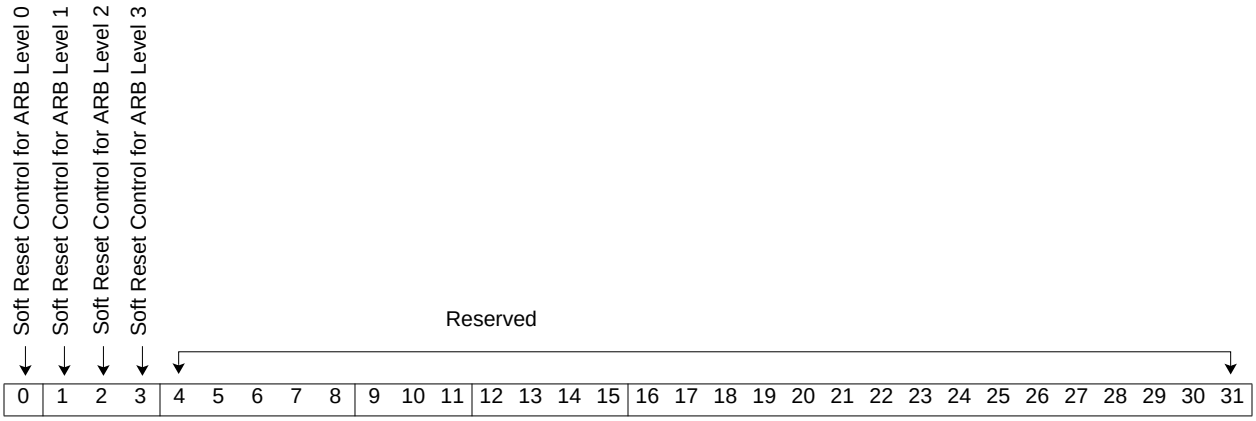
CPU Soft Reset Register

This register provides software with a mechanism to issue soft resets to each of the processors. When the CPC710 detects a write to this register, the corresponding $\overline{\text{SYS\_SRESET}}$ signal is driven active for a minimum time depending on the setting of bit 10 to 15 of the RTBR register (see *“CPC0_RTBR” on page 9-33*).

Reset Value x’0000 0000’

Address x‘FF00 1040’

Access Type Write Only


Soft Reset Control for ARB Level 0
Soft Reset Control for ARB Level 1
Soft Reset Control for ARB Level 2
Soft Reset Control for ARB Level 3
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

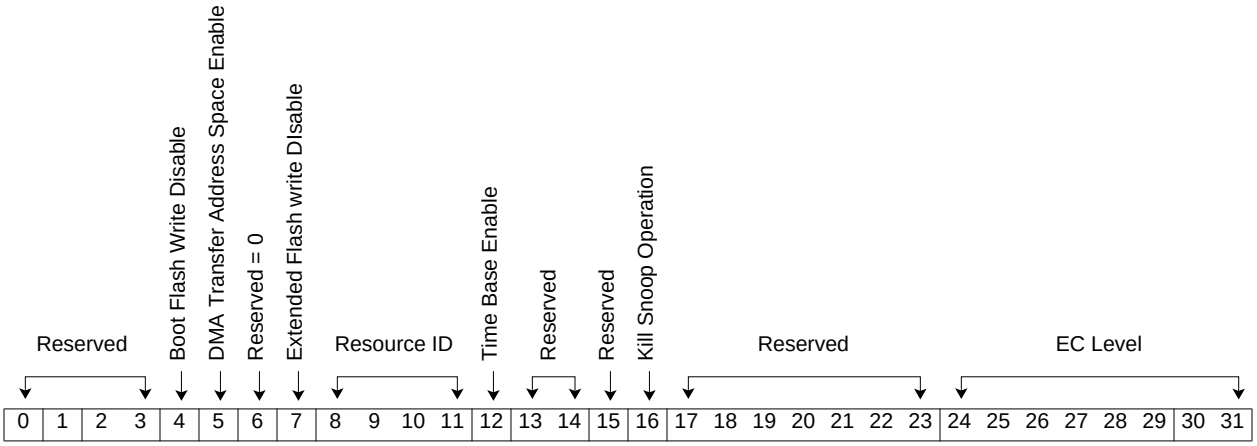
Bit(s)	Description
0	**Soft Reset Control for ARB Level 0** 0: Writing 0 to this bit has no effect 1: Writing 1 to this bit will initiate a pulse on the $\overline{\text{SYS_SRESET}}$0 signal.
1	**Soft Reset Control for ARB Level 1** 0: Writing 0 to this bit has no effect 1: Writing 1 to this bit will initiate a pulse on the $\overline{\text{SYS_SRESET}}$1 signal.
2	**Soft Reset Control for ARB Level 2** 0: Writing 0 to this bit has no effect 1: Writing 1 to this bit will initiate a pulse on the $\overline{\text{SYS_SRESET}}$2 signal.
3	**Soft Reset Control for ARB Level 3** 0: Writing 0 to this bit has no effect 1: Writing 1 to this bit will initiate a pulse on the $\overline{\text{SYS_SRESET}}$3 signal.
6-31	**Reserved**

This register is used to enable address ranges to be decoded by the CPC710 and processor-related operations.

Reset Value x'0008 00A0

Address x'FF00 1000'

Access Type Read/Write


Reserved
Boot Flash Write Disable
DMA Transfer Address Space Enable
Reserved = 0
Extended Flash write DIsable
Resource ID
Time Base Enable
Reserved
Reserved
Kill Snoop Operation
Reserved
EC Level
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0-3	**Reserved**
4	**Boot Flash Write Disable** 0: Indicates that writes are allowed to Boot FLASH space 1: Writes to Boot FLASH space are inhibited and an error is generated
5	**DMA Transfer Address Space Enable** 0: Accesses to DMA Address Range allowed 1: Accesses to DMA Address Range inhibited
6	**Reserved**. R/W Must be left to 0
7	**Extended Flash Write Disable** 0: Indicates that writes are allowed to extended FLASH space 1: Writes to extended FLASH space are inhibited
8 - 11	**Resource ID** This 4-bit field contains the Resource ID that device uses to determine whether or not it is the target of a DMA transfer operation initiated by a ecowx or eciwx operation. The Resource ID is encoded on the SYS_TBST and SYS_TSIZ[0:2] signals during this cycle.
12	**Time Base Enable** 0: The Time Base Enable signal to the CPU is deactivated. CPU real time clocks halted 1: The Time Base Enable signal SYS_TBE to the CPU is activated. CPU real time clocks enabled
13 - 14	**Reserved**
15	**Reserved** R/W

CPC0_UCTL

Universal System Control

Bit(s)	Description
16	**Kill Snoop Operation (Must set to 0 for the PowerPC 750)** 0: The CPC710 issues Kill address only transaction types for full cache line invalidates 1: The CPC710 issues Flush address only transaction type for full cache line invalidates (Workaround of the 604 errata "Kill snoop bug")
17 - 23	**Reserved**
24 - 31	**CPC710 - EC LEVEL**. Read only Bit 24 Always at one Bits 25 to 27 Main Engineering changes Bits 28 to 31 Sub Engineering changes b'1000 0000 x'80 for CPC710_100 b'1001 0000 x'90 for CPC710_100+ b'1010 0000 x'A0 for CPC710_133

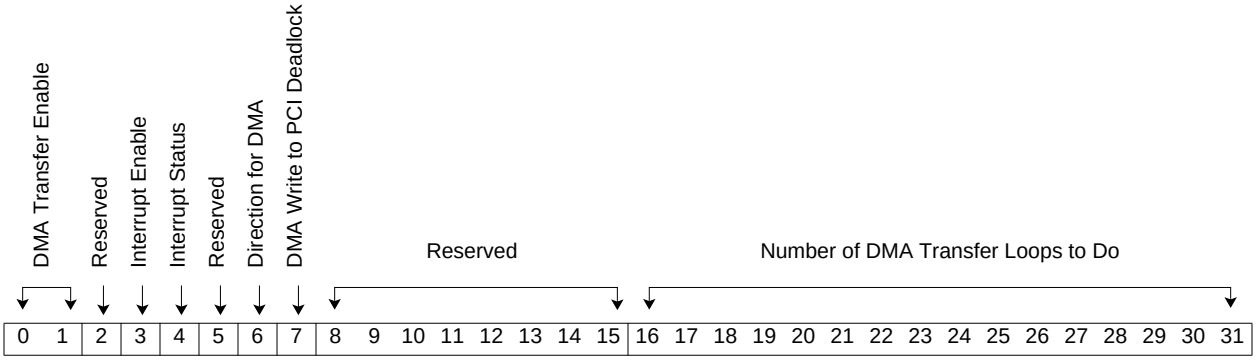
DMA0_GSCRx

DMA Global Control Register

Reset Value x'0000 0000'

Address User x'FF1C 0020'
Privileged x'FF1E 0020'

Access Type User Read Only
Privileged Read/Write


DMA Transfer Enable
Reserved
Interrupt Enable
Interrupt Status
Reserved
Direction for DMA
DMA Write to PCI Deadlock
Reserved
Number of DMA Transfer Loops to Do
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**DMA Transfer Enable** 00: Reset DMA Controller to default power up mode. 10: DMA Controller chained mode suspended (The Chained mode is back after writing 11) 11: DMA Controller is enabled
2	**Reserved.** R/W
3	**Interrupt Enable** When set, generates an interrupt at the completion of a DMA transfer 0: IT2 disabled 1: IT2 enabled
4	I**nterrupt Status** 0: End of DMA transfer interrupt IT2 not asserted 1: End of DMA transfer interrupt IT2 asserted Software must write a 0 to Reset the IT2 Interrupt
5	**Reserved.** R/W
6	**Direction for DMA** 0: PCI to MEMORY 1: MEMORY to PCI
7	**Deadlock Avoidance with DMA Write to PCI** 0: Disabled 1: Enabled Set this bit to 1 to avoid bus hangs on PCI interface
8 - 15	**Reserved.** RO

DMA0_GSCRx

DMA Global Control Register

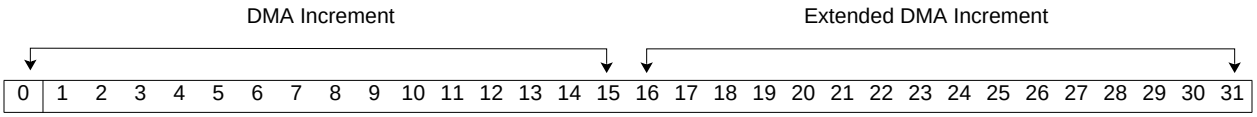
Bit(s)	Description
16 - 31	**Number of DMA Transfer Loops to Do**. The Data Byte length transferred in a Loop is defined by bit[3:15] of the DMA0_XSCRx register. If not equal to 0, these bits specify the number of loops to do. During an Extended DMA, these bits contain the number of loops remaining. If the DMA is initiated by a write to the DMA0_XTAR register, set these bits to x'0001'. During an Extended DMA, contains the number of loops remaining.

DMA0_XCLRx

DMA Cache Line Increment Register

Reset Value x'0000 0000

Address	User	x'FF1C 0030'
	Privileged	x'FF1E 0030'
Access Type	User	Read/Write
	Privileged	Read/Write

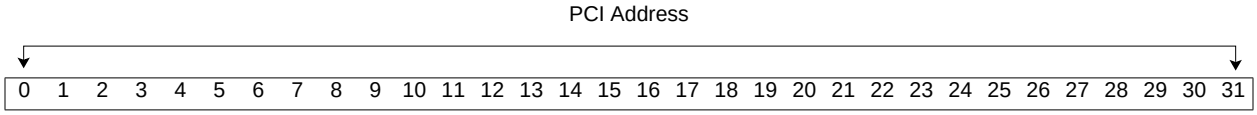


Bit(s)	Description
0-15	**DMA Increment** This field specifies cache line increment associated with each successive cache line transfer
16 - 31	**Extended DMA Increment** This field specifies cache line increment between successive extended DMA

DMA0_XPARx

DMA Transfer PCI Address Register

Reset Value	x'0000 0000'	
Address	User	x'FF1C 0070'
	Privileged	x'FF1E 0070'
Access Type	User	[0:3] Read Only, [4:31] Read/Write
	Privileged	[0:3] Read/Write, [4:31] Read Only

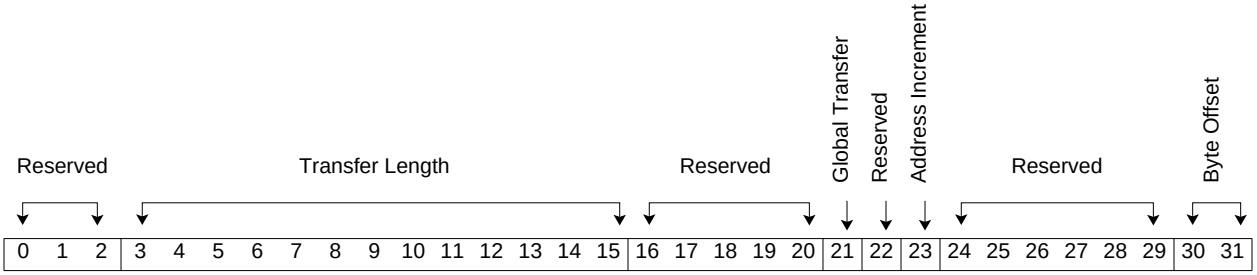


Bit(s)	Description
0 - 31	**PCI Address (CPU)**. Contains the adapter I/O address for the DMA transfer operation.

Reset Value x'0000 0000'

Address User x'FF1C 0040'
Privileged x'FF1E 0040'

Access Type Read/Write
(User and Privileged)


Reserved
Transfer Length
Reserved
Global Transfer
Reserved
Address Increment
Reserved
Byte Offset
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 2	**Reserved**
3 - 15	**Transfer Length.** Contains the number of Bytes to be transferred in a Loop (maximum is 4 K). A value of 0 will transfer 0 bytes.
16 - 20	**Reserved**
21	**Global Transfer** 0: No snoop operations required for accesses to system memory 1: Accesses to system must be coherent
22	**Reserved**
23	**Address Increment** 0: Do NOT increment I/O address during DMA transfer 1: Increment I/O address during DMA transfer
24 - 29	**Reserved**
30 - 31	**Byte Offset**. Specifies the byte offset associated with the DMA transfer real address

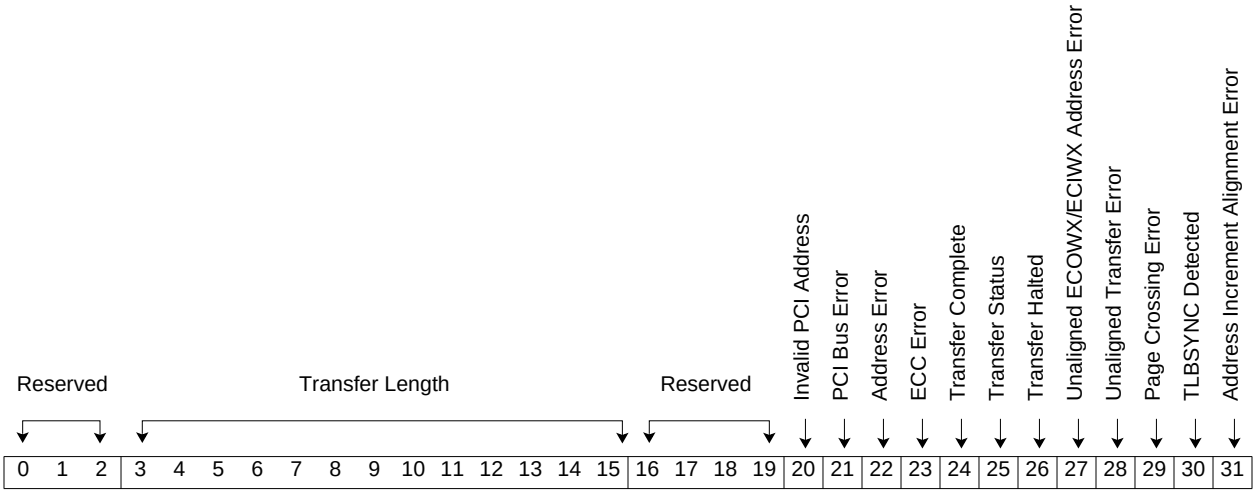
DMA0_XSSRx

DMA Transfer Status Register

Reset Value x'0000 0000'

Address User x'FF1C 0050'
Privileged x'FF1E 0050'

Access Type Read Only
(User and Privileged)


Reserved
Transfer Length
Reserved
Invalid PCI Address
PCI Bus Error
Address Error
ECC Error
Transfer Complete
Transfer Status
Transfer Halted
Unaligned ECOWX/ECIWX Address Error
Unaligned Transfer Error
Page Crossing Error
TLBSYNC Detected
Address Increment Alignment Error
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 2	**Reserved**
3 - 15	**Transfer Length**. Contains the number of bytes remaining when the transfer was completed or aborted.
16 - 19	**Reserved**
20	**Invalid PCI Address** 1: DMA0_XPARx did not match any PCI extents
21	**PCI Bus Error** 1: Error detected during PCI bus transaction.
22	**Address Error** 1: Invalid memory address detected
23	**ECC Error** 1: Double-bit ECC error detected in memory
24	**Transfer Complete** 0: Transfer is not complete 1: Transfer complete in Normal DMA mode of operation. For the Extended Mode and the Chained mode, the end of the DMA transfer is indicated only by the activation of the $\overline{IT2}$ interrupt signal and DMA0_GSCRx[4] (interrupt status bit).
25	**Transfer Status** 0: No DMA transfer in progress 1: DMA transfer operation is underway

Bit(s)	Description
26	**Transfer Halted** 1: DMA transfer operation in progress was halted due to start of second DMA transfer operation
27	**Unaligned ECOWX/ECIWX Address Error** 1: Address associated with ECOWX/ECIWX is not word (32-bit) aligned
28	**Unaligned Transfer Error** 1: Address alignment error when the XTAR address (memory) is not doubleword (64-bit) aligned with XPAR address (PCI).
29	**Page Crossing Error** 1: Page (4KB) Crossing detected during DMA transfer
30	**TLBSYNC Detected** 0: No TLBSYNC Detected 1: TLBSYNC detected during DMA transfer Transfer
31	**Address Increment Alignment Error** 1: Improper alignment of addresses when Address Increment bit is off. XTAR address not doubleword-aligned and XPAR address not cacheline-aligned.

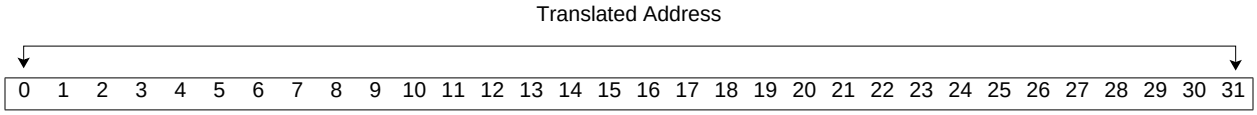
DMA0_XTARx

DMA Transfer Translated Address Register

Reset Value x'0000 0000'

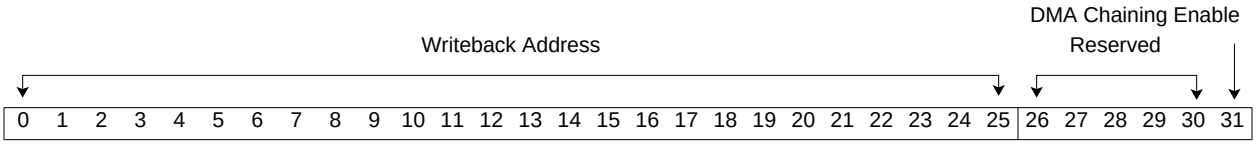
Address User x'FF1C 00A0'
Privileged x'FF1E 00A0'

Access Type Read/Write
(User and Privileged)



Bit(s)	Description
0 - 31	**Translated Address.** Contains the 32-bit real address presented on the processor bus during the ecowx/eciwx transfer. A write to this register will start the DMA operation.

Reset Value	x'0000 0000'	
Address	User	x'FF1C 0090'
	Privileged	x'FF1E 0090'
Access Type	User	Read Only
	Privileged	Read/Write



Bit(s)	Description
0-26	**Writeback/Readback Address.** This register contains the real address used by the CPC710 to read its next buffer descriptor in memory, at the end of the DMA transfer operation. For the last buffer description this register contains the real address used by the CPC710 to write its completion status.
27 - 30	**Reserved** (assumed to be zero).
31	**DMA Chaining Enable** When set to 1, it indicates that the DMA chaining is enabled, and all registers for the next DMA will be loaded with the values stored in memory (descriptor) at the address given by bits 0-26.

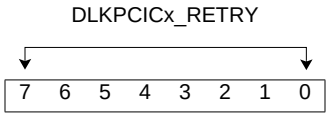
PCIC0_DLKRETRY

PCI32 DeadLock Retry Counter

Reset Value x'00'

Address x'51'

Access Type Read/Write



Bit(s)	Description
7 - 0	Available only for the PCI32, this 8-bit counter is used to limit the number of Retries in the case of an access in a deadlock area space defined with the PCILx_BPMDLK/PCILx_TPMDLK or PCILx_BIODLK/PCILx_TIODLK registers.

PCIC1_INTRESET

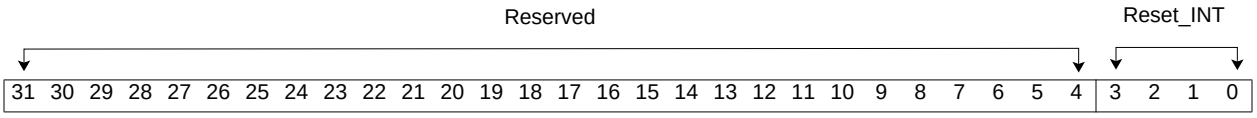
PCI64 Reset Interrupt

Resets one of the posted interrupt $\overline{\text{G\_INTA}}$, $\overline{\text{G\_INTB}}$, $\overline{\text{G\_INTC}}$, $\overline{\text{G\_INTD}}$ on the PCI-64bit bus. Reset can be done from the PCI-64 or from the CPU in configuration mode. The CPU can only execute the SET of INTA, INTB, INTC, INTD when writing in Register INT_SET at address CPC0_PCIBAR + x'000F 8310'.

Reset Value x'0000 0000'

Address x'68'

Access Type Read/Write



Bit(s)	Description
31 - 4	**Reserved**
3 - 0	**RESET_Interrupts** 0: No action 1: Resets the bit corresponding to one PCI64 interrupt Bit 0: $\overline{\text{G\_INTA}}$ Bit 1: $\overline{\text{G\_INTB}}$ Bit 2: $\overline{\text{G\_INTC}}$ Bit 3: $\overline{\text{G\_INTD}}$

PCIC1_ITADDSET

Set PCI64 Interprocessor Interrupt

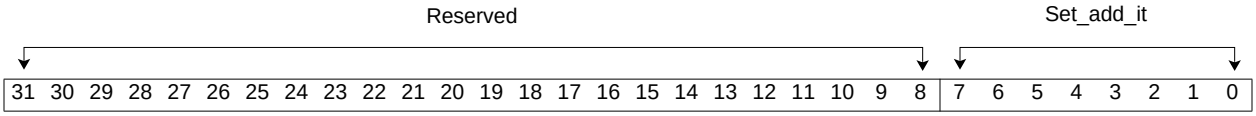
This is a Virtual Register. When addressed, the interrupt signal $\overline{INT}$1 is set (goes to 0). The SET can be done from the PCI-64 or from the PowerPC CPU in configuration mode.

Only the PowerPC CPU can reset the interrupt $\overline{INT}$1 by writing a "1" in the PCIL1_ITADDRESET interrupt reset register.

Reset Value x'0000 0000'

Address x'64'

Access Type Write Only



Bit(s)	Description
31 - 8	**Reserved**
7 - 0	**Set_add_it** 1: Writing a 1 in one of these 8 bits SETS the interrupt signal $\overline{INT}$1 0: No action

PCIC1_PPBAR

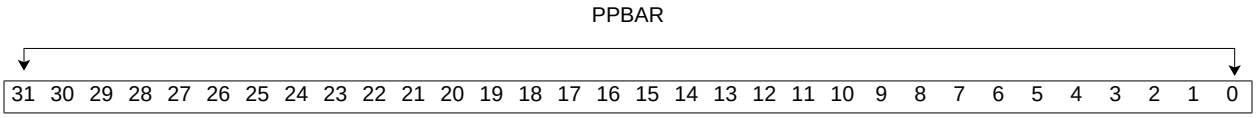
PCI64 Base Address for PCI to System Extended Access

Only the PCI64 bridge has this register at these location. For PCI32 bridge this register is in the Specific PCI32 PCI Bridge Space (see page 9-8).
This register must be used if extended memory space is used (see bit 27 of PGCHP register) and if FPHB mode is selected (this register as no action in CHRP mode or PREP mode).

Reset Value x'0000 0000'

Address x'14'

Access Type Read/Write



Bit(s)	Description
31 - 0	**PCI Base Address.** Contains upper bits of the System Base address that memory is mapped to.

PCIC1_PSBAR

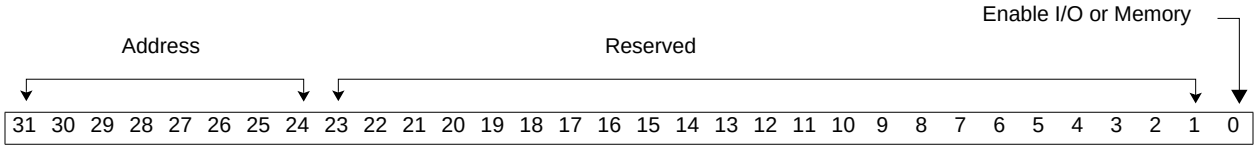
PCI64 Base Address for PCI to System Access

Only the PCI64 bridge has this register at this location. For PCI32 bridge this register is in the Specific PCI32 PCI Bridge Space (see page 9-8).

Reset Value x'0000 0000'

Address x'10'

Access Type Read/Write

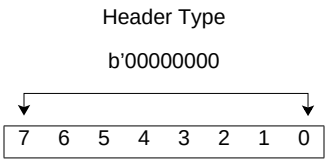


Bit(s)	Description
31 - 24	**PCI Base Address** Contains the upper bits of the System Base address that memory is mapped to.
23 - 1	Reserved
0	**Enable Memory or IO Space (**copy of the bit 7 of the PCILx_PSSIZE Register) 0: Memory Space 1: IO Space

Reset Value x’00’

Address x‘0E’

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies the layout of bytes x’10’ through x’3F’ in the configuration header and whether or not a particular device contains multiple functions. The CPC710 always responds with x’00’ to reads to indicate Layout 0. Writes to this register are ignored.

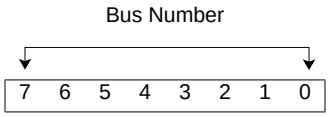
PCICx_BUSNO

Bus Number

Reset Value x'00'

Address x'40'

Access Type Read/Write

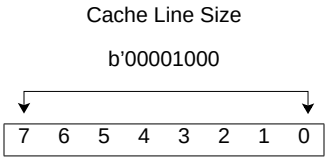


Bit(s)	Description
7 - 0	Contains the assigned bus number for this bridge. The CPC710 uses this number to determine what action to take for configuration cycles directed to this bridge. After reset, this register contains a value of x'00'.

Reset Value x’08’

Address x‘0C’

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies the cache line size in units of 32-bit words. The CPC710 always responds with x’08’ for reads to indicate that device will always disconnect from any PCI master burst operation that crosses a 32-byte boundary.

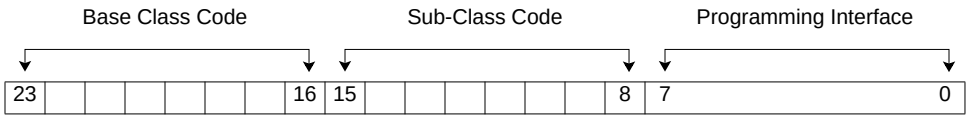
PCICx_CLS

PCI Class Register

Reset Values x'060000'

Address x'09'

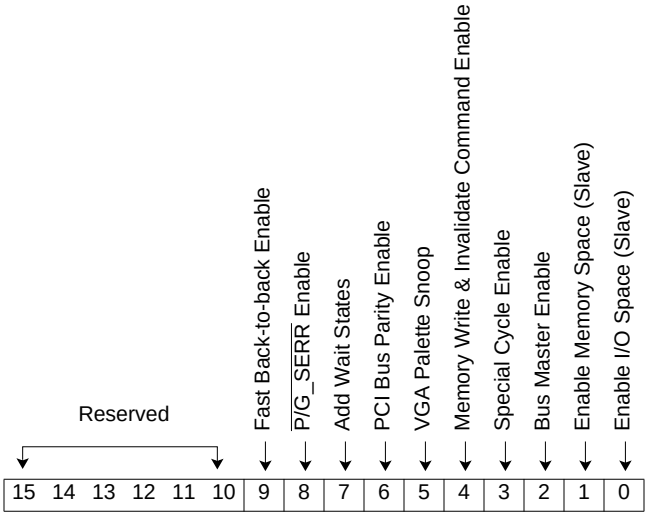
Access Type Read Only



Bit(s)	Description
23-16	Classifies the type of function this device performs. The CPC710 always responds with x'06' for reads to indicate a Bridge device.
15-8	Specifically identifies a particular function of the Base Class Code register. The CPC710 always responds with x'00' for reads to indicate a HOST type of bridge device.
7 - 0	Defines a specific register-level programming interface. The CPC710 always responds with x'00' for reads from this register.

This register provides control over PCI bridge behavior.

Reset Value	PCI32	x'0000'
	PCI64	x'0000'
programming example	PCI32	x'0156'
	PCI64	x'0156'
Address	x'04'	
Access Type	Read/Write	


Reserved
Fast Back-to-back Enable
P/G_SERR Enable
Add Wait States
PCI Bus Parity Enable
VGA Palette Snoop
Memory Write & Invalidate Command Enable
Special Cycle Enable
Bus Master Enable
Enable Memory Space (Slave)
Enable I/O Space (Slave)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Bit(s)	Description
15 - 10	**Reserved**
9	**Fast Back-to-back Enable** 0: Disabled 1: PCI Bridge issues fast back-to-back transfers without regard to which target is being addressed, providing that the previous transaction was a write. Note: This bit should be set if all slaves on the PCI bus support this capability.
8	**SERR Enable** 0: PCI Bridge will not assert $\overline{\text{P/G\_SERR}}$ upon detecting an error. 1: PCI bridge will assert $\overline{\text{P/G\_SERR}}$ for PCI address parity error
7	**Add Wait States**. Read Only. Always returns 0. Device does not support address data stepping.
6	**PCI Bus Parity Enable** 0: Device will disable all parity checking on the PCI bus 1: Device will detect and report parity errors on the PCI bus
5	**VGA Palette Snoop**. Read Only. Always returns 0. Device is not VGA compatible.
4	**Memory Write & Invalidate Command Enable** 0: Device does not generate this type of cycle. 32-byte transfers use the Memory Write command. 1: Device generates this cycle as a master for any 32-byte transfer.

PCICx_CMD

PCI Command

Bit(s)	Description
3	**Special Cycle Enable**. Read Only. Always returns 0. Device will not respond to Special Cycle commands.
2	**Bus Master Enable** 0: PCI Bridge master capability is disabled. 1: PCI Bridge performs as a PCI master for accesses to its address spaces.
1	**Enable Memory Space (Slave)** 0: PCI Bridge will not respond to memory accesses on the PCI bus 1: PCI Bridge will respond to memory accesses on the PCI bus
0	**Enable I/O Space (Slave)** 0: PCI Bridge will not respond to IO accesses on the PCI bus 1: PCI Bridge will respond to IO accesses on the PCI bus except for PREP mode (see note below)

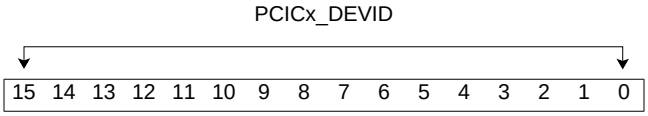
Note: I/O cycles for Slave as defined in bit 0 are not decoded by the CPC710 when the address mapping is in PREP mode. See PREP mode definition.

PCICx_DEVID

Device ID

This register identifies a particular device.

Reset Value	PCI32	x'0105'
	PCI64	x'00FC'
Address	x'02'	
Access Type	Read Only	



Bit(s)	Description
15 - 0	**Device Identification Number** **PCI32:** Value: x'0105' (x'05' for address 02 and x'01' for address 03) **PCI64:** Value: x'00FC' (x'FC' for address 02 and x'00' for address 03)

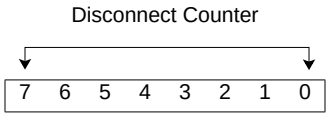
PCICx_DISCNT

Disconnect Register

Reset Value x'00'

Address x'42'

Access Type Read/Write

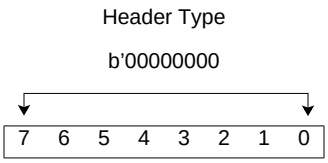


Bit(s)	Description
7 - 0	The CPC710 uses this register when acting as a target device as a time-out mechanism in burst operations. The value written to this register is multiplied by four and used to determine when the bridge should assert STOP#. After reset, this register contains x'00' which disables the timer. This counter is enabled only if bit 0 (for PCI 32) or bit 8 (for PCI-64) of CPC0_AVDG Register is set (see "CPC0_AVDG" on page 9-15) When time-out occurs the bit 9 of the PCILx_CSR Register is set (see "PCILx_CSR" on page 9-91).

Reset Value x’00’

Address x‘0E’

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies the layout of bytes x’10’ through x’3F’ in the configuration header and whether or not a particular device contains multiple functions. The CPC710 always responds with x’00’ to reads to indicate Layout 0. Writes to this register are ignored.

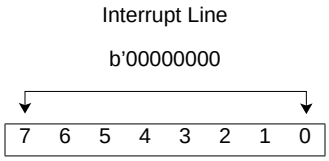
PCICx_INTLN

Interrupt Line

Reset Value x’00’

Address x‘3C’

Access Type Read Only

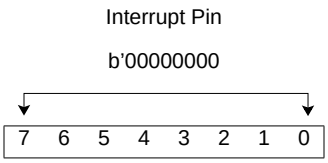


Bit(s)	Description
7 - 0	Indicates interrupt routing information for devices that implement an interrupt. The PCI bridge logic does not generate interrupts and therefore this register is not implemented. The CPC710 responds with x’00’ to reads from this register and ignores Writes.

Reset Value x’00’

Address x‘3D’

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies which particular interrupt pin, INTA, INTB, INTC, or INTD, is used to generate interrupts. Since the PCI bridge does not generate any interrupts, the CPC710 responds with x’00’ to reads from this register and ignores writes.

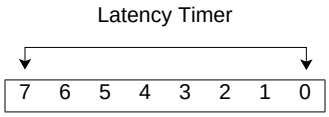
PCICx_LATTIM

Latency Timer

Reset Value x’00’

Address x‘0D’

Access Type Read/Write

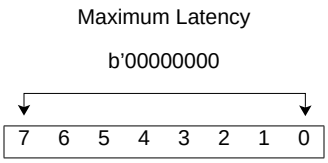


Bit(s)	Description
7 - 0	Provides bus masters with a minimum guaranteed time slice on the PCI bus. The value programmed into this register is the minimum number of PCI bus clocks that a master can own the PCI bus starting from the cycle that FRAME is activated. This register is set to X’00’ at reset. Maximum number of PCI bus clocks 128

Reset Value x’00’

Address x‘3F’

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies how often the device needs to gain access to the PCI bus in 0.25 µs. The CPC710 has no specific requirements and therefore always responds with x’00’.

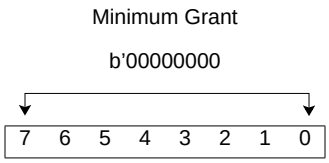
PCICx_MINGNT

Minimum Grant

Reset Value x'00'

Address x'3E'

Access Type Read Only



Bit(s)	Description
7 - 0	Specifies the length of a device's burst period in 0.25 μsecs. The CPC710 has no specific requirements and therefore always responds with x'00'.

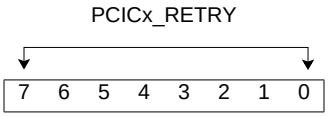
PCICx_RETRY

Retry Counter

Reset Value x’00’

Address x‘50’

Access Type Read/Write



Bit(s)	Description
7 - 0	When the device is a Master on the PCI bus, this register is used as a time-out mechanism for continuous retries on the PCI bus. Whenever a retry occurs for a particular address, the PCI bridge logic increments (decrements) a counter. The 8-bit counter is reset whenever data is transferred. If the count reaches the value specified in this register, the PCI bridge logic will not retry the access and will report the Result by writing bit 5 of the PCILx_PLSSR Register (see “PCILx_PLSSR” on page 9-101). After reset, the register contains x’00’ which disables the retry counter.

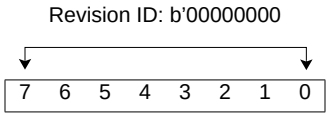
PCICx_REVID

Revision ID

Reset Value x’03’

Address x‘08’

Access Type Read Only



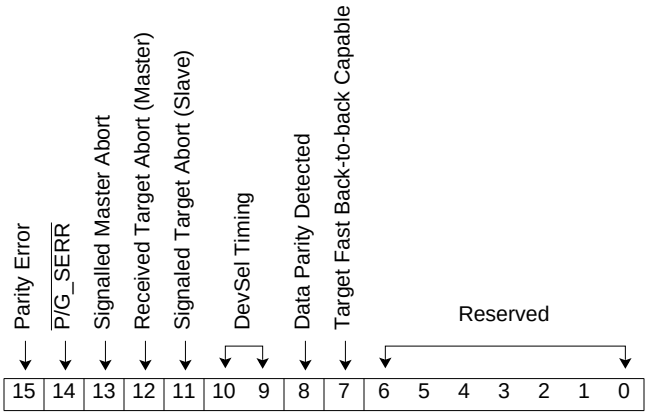
Bit(s)	Description
7 - 0	Provides an extension to the PCI Device ID register. The CPC710 always responds with x’03’ for reads from this register.

This register records status and error information from PCI bus transfers. Reads from this register behave normally. Writes to this register are restricted, in that software cannot set any bit in this register, only reset. Additionally, to reset a bit, software must write a 1 to the corresponding bit location. For example, to reset only bit 14, software must write '0100 0000 0000 0000'b to this register.

Reset Value x'0280'

Address x'06'

Access Type Read/Write


Parity Error
P/G_SERR
Signalled Master Abort
Received Target Abort (Master)
Signaled Target Abort (Slave)
DevSel Timing
Data Parity Detected
Target Fast Back-to-back Capable
Reserved
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Bit(s)	Description
15	**Parity Error** 0: No Error 1: PCI Bridge has detected a parity error; bit set even if parity checking is disabled.
14	**Signaled System Error (P/G_SERR)** 0: No Error 1: PCI Bridge has asserted SERR due to an address parity error.
13	**Signaled Master Abort** 0: No Error 1: PCI Bridge has issued a master abort.
12	**Received Target Abort (Master)** 0: No Error 1: PCI Bridge has detected a target abort for one of its transactions.
11	**Signaled Target Abort (Slave)** 0: No Error 1: PCI Bridge as a slave has issued a target abort.
10 - 9	**DevSel Timing**. Read Only 01: PCI Bridge responds with Medium timing on P/G_DEVSEL signal.

PCICx_STATUS

PCI Status

Bit(s)	Description
8	**Data Parity Detected** 0: No Error 1: This bit is set if the following 3 conditions are met: I) PCI Bridge asserted, or observed P/G_PERR signal on PCI bus II) PCI Bridge acting as master III) Bit 6 of Command Register set
7	**Target Fast Back-to-back Capable**. Read Only. Always returns a 1 to indicate that the PCI Bridge as a target will accept fast back-to-back transfers when the transfers are not to the same device.
6 - 0	**Reserved**

Reset Value x’00’

Address x‘41’

Access Type Read/Write

Subordinate Bus Number

7	6	5	4	3	2	1	0

Bit(s)	Description
7 - 0	Specifies the largest bus number beneath this bridge. After reset, this register contains a value of x’00’.

PCICx_VENDID

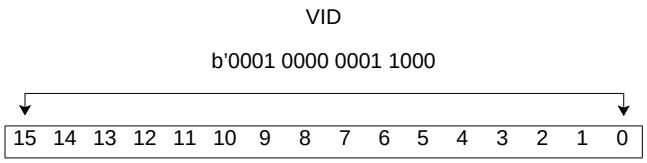
Vendor ID

This register identifies the device manufacturer.

Reset Value x’1014’

Address x‘00’

Access Type Read Only



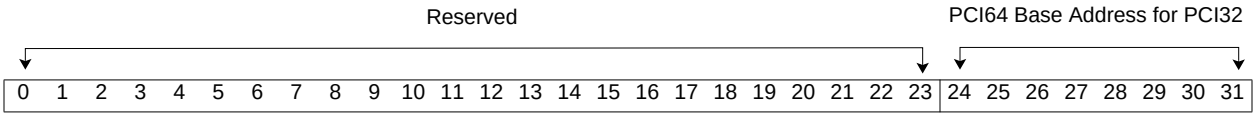
Bit(s)	Description
15 - 0	**Vendor Identification Number** Value = x’1014’; (x’14’ for address 00 and x’10’ for address 01)

Only the PCI32 bit bridge has this register at these location. For PCI64 bit bridge this register is in “Standard PCI Configuration Space” on page 9-5.
This register must be used if extended memory space is used (see bit 27 of CPC0_ PGCHP register) **and if FPHB mode is selected** (this register has no action in CHRP mode or PREP mode).

Reset Value x’0000 0000’

Address CPC0_PCIBAR + x’000F 8150’

Access Type Read/Write



Bit(s)	Description
0 - 23	**Reserved**
24 - 31	**PCI Base Address.** Contains the upper bits of the PCI Base address that PCI is mapped to.

PCIL0_PSBAR

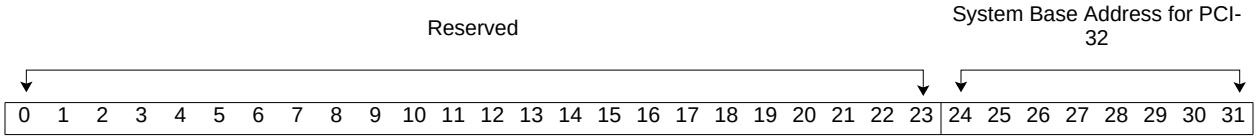
PCI32 Base Address for PCI to System Access

Only the PCI32 bit bridge has this register at these location. For PCI64 bit bridge this register is in “Standard PCI Configuration Space” on page 9-5.

Reset Value x’0000 0000’

Address CPC0_PCIBAR + x’000F 8140’

Access Type Read/Write



Bit(s)	Description
0 - 23	**Reserved**
24 - 31	**PCI Base Address.** Contains the upper bits of the PCI Base address that PCI is mapped to.

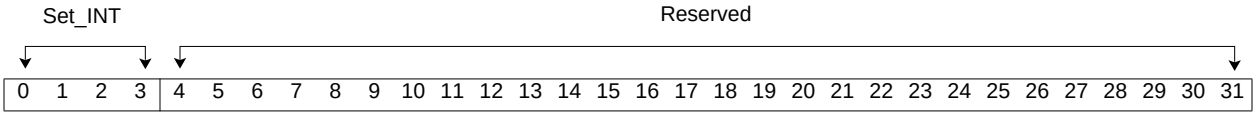
PCIL1_INTSET

Set PCI64 Interrupt Register

This register exists only for PCI64. Interrupt can be set only by the CPU.

Address CPC0_PCIBAR + ’000F 8310’

Access Type Write Only



Bit(s)	Description
0 - 3	**Set_It** Writing 1 by the CPU set the bit corresponding Writing 0 has no action Bit 0: $\overline{\text{INTA}}$ Bit 1: $\overline{\text{INTB}}$ Bit 2: $\overline{\text{INTC}}$ Bit 3: $\overline{\text{INTD}}$
4 - 31	**Reserved**

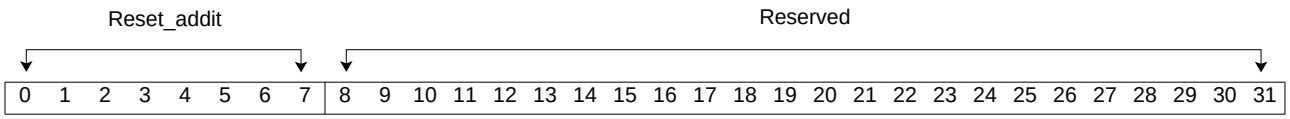
PCIL1_ITADDRESET

Reset Addressed Interrupt Register

This Virtual register exists only for PCI64 bridge. Only the CPU can write to this register and reset the $\overline{IT}$1 output interrupt signal.

Address CPC0_PCIBAR + ’000F 8300’

Access Type Write Only



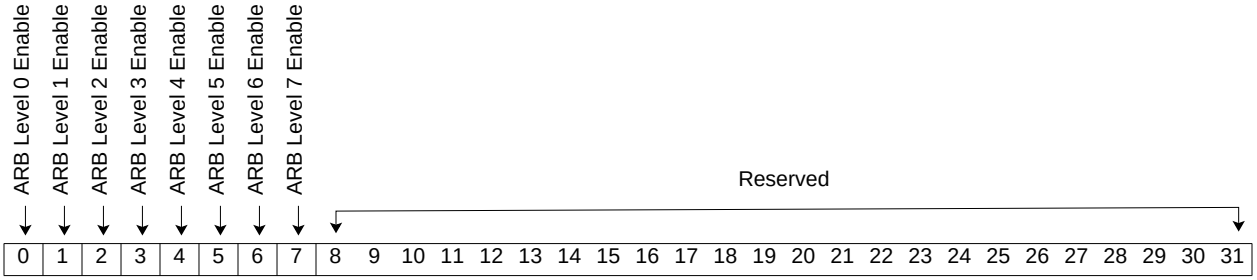
Bit(s)	Description
0 - 7	**Reset_addit** 1: Writing a 1 in one of these 8 bits resets the interrupt signal $\overline{IT}$1 0: No action
8 - 31	**Reserved**

This register provides software with a means to disable individual devices on the PCI bus from generating master bus operations.

Reset Value x'0000 0000'

Address CPC0_PBIBAR + x'000F 7F30'

Access Type Read/Write


ARB Level 0 Enable
ARB Level 1 Enable
ARB Level 2 Enable
ARB Level 3 Enable
ARB Level 4 Enable
ARB Level 5 Enable
ARB Level 6 Enable
ARB Level 7 Enable
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**ARB Level 0 Enable (correspond to signal pair G_REQ0, G_GNT0 on PCI64 or P_REQ0, P_GNT0 on PCI32)** 0: ARB level is ignored 1: ARB level is enabled
1	**ARB Level 1 Enable**
2	**ARB Level 2 Enable**
3	**ARB Level 3 Enable**
4	**ARB Level 4 Enable**
5	**ARB Level 5 Enable**
6	**ARB Level 6 Enable**
7	**ARB Level 7 Enable** (Not supported in 32-bit PCI bridge)
8-31	**Reserved**

PCILx_BARPP

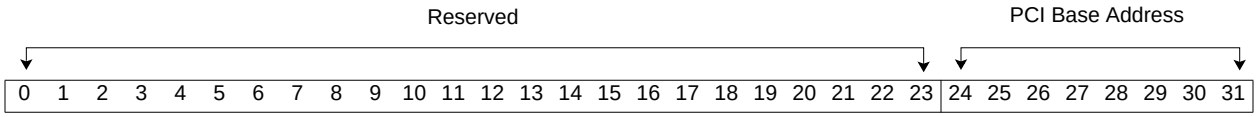
System Base Address for PCI to System Extended Access

The definition is the same for 32-bit PCI and 64-bit PCI. This register must be used if extended memory space is used (see bit 27 of CPC0_PGCHP register) **and if FPHB mode is selected** (this register as no action in CHRP mode or PREP mode).

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8130'

Access Type Read/Write



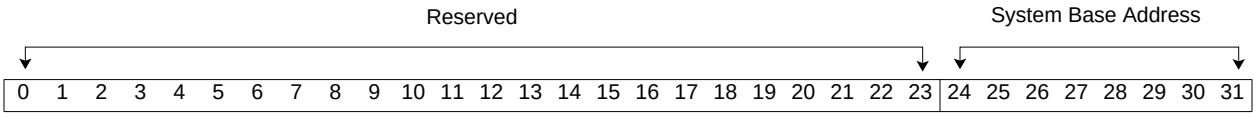
Bit(s)	Description
0 - 23	**Reserved.**
24 - 31	**System Base Address**. Contains the upper bits of the PCI Base address that PCI is mapped to.

The definition is the same for 32-bit PCI and 64-bit PCI.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8120'

Access Type Read/Write



Bit(s)	Description
0 - 23	**Reserved**
24 - 31	**System Base Address.** Contains the upper bits of the system Base address that Memory is mapped to.

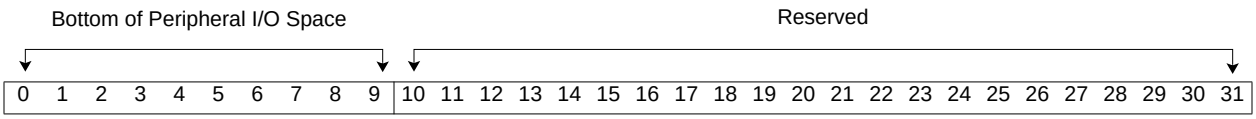
PCILx_BIODLK

Bottom of Peripheral I/O Space with Deadlock Avoidance

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8220'

Access Type Read/Write

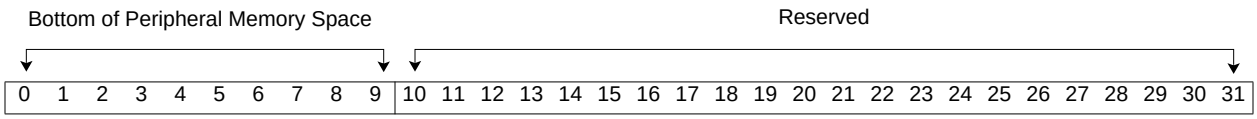


Bit(s)	Description
0 - 9	**Bottom of Peripheral IO Space**. Contains the bottom address for the CPU to PCI IO access with potential deadlock
10 - 31	**Reserved**

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8200'

Access Type Read/Write



Bit(s)	Description
0 - 9	**Bottom of Peripheral Memory Space**. Contains the bottom address for the CPU to PCI MEMORY access with potential deadlock
10 - 31	**Reserved**

PCILx_CFGADDR

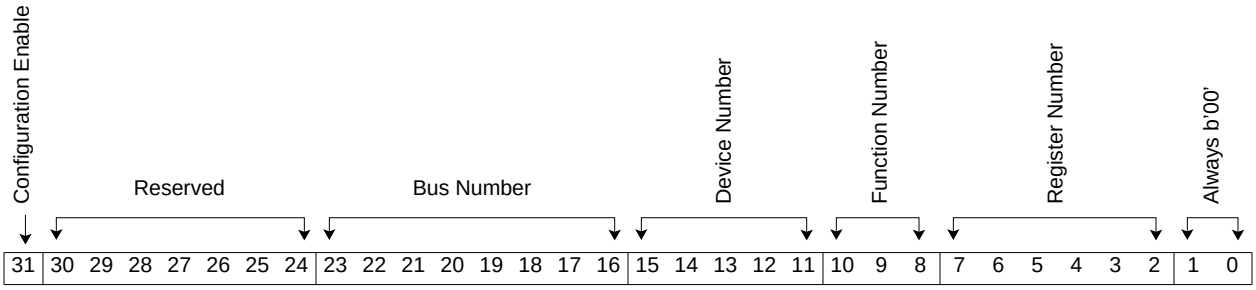
Configuration Address

This Little Endian register, along with the PCICx_CFGDATA register, provides software with a means to configure the PCI bus. The CPC710 implements Configuration Mechanism #1 as specified in the PCI Local Bus Specification. See heading “Configuration Cycles” on page 7-4 for additional details.

Reset Value x’0000 0000’

Address CPC0_PCIBAR + x’000F 8000’

Access Type Read/Write


Configuration Enable
Reserved
Bus Number
Device Number
Function Number
Register Number
Always b'00'
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Bit(s)	Description
31	**Configuration Enable** 0: Disabled 1: Enabled, accesses to the CPC0_CFGDATA register result in CPC710 executing a configuration access to itself or to the PCI bus.
30 - 24	**Reserved**
23 - 16	**Bus Number.** Specifies which PCI bus is being configured. The CPC710 checks this field to determine the appropriate configuration action.
15-11	**Device Number**. Selects a particular device to be configured on a PCI bus. Device # 0 is the CPC710
10-8	**Function Number.** For devices that implement more than one function, this field specifies which function to configure within a device. 0000 : Access to the CPC710 PCI Configuration Space registers 1111 : Special Cycle command issue to the PCI bus device #31
7-2	**Register Number**. Specifies which register out of the 256-byte PCI Configuration header to access
1-0	**Always b’00’**

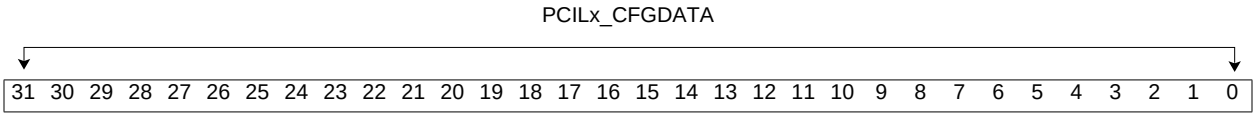
PCILx_CFGDATA

Configuration Data Register

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8010'

Access Type Read/Write



Bit(s)	Description
31 - 0	**PCI Configuration Data - Virtual 32-bit Register.** When this Little Endian register is accessed in Read or Write, the CPC710 initiates a PCI Configuration Read or Write cycle of external PCI devices, the address of which is provided by PCICx_CFGADDR.

PCILx_CRR

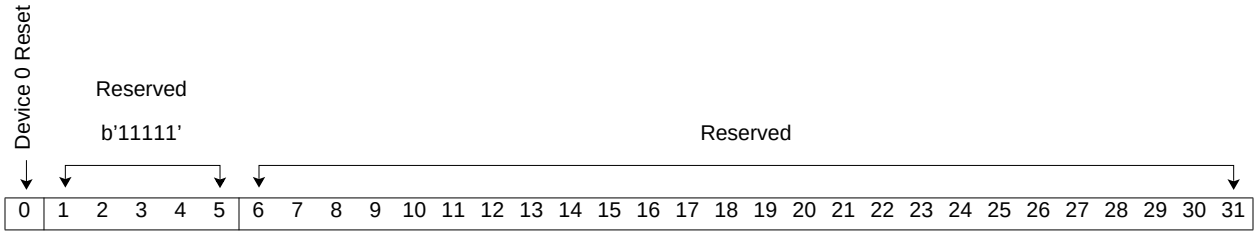
Component Reset Register

This register provides software with a means to disable all devices on the PCI bus by writing a zero in bit 0.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 7EF0'

Access Type Read/Write



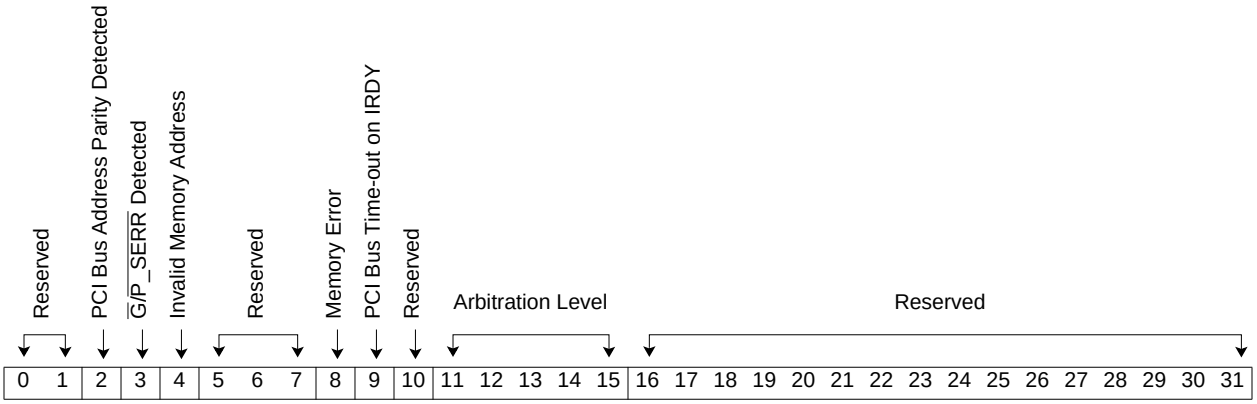
Bit(s)	Description
0	**Device Reset** 0: Reset signal $\overline{\text{P\_RST}}$ or $\overline{\text{G\_RST}}$ is active 1: Reset signal inactive
1 - 5	**Reserved**. Must be left to 1
6 - 31	**Reserved**.

This register is used to log errors during PCI Master to system transfers. See “PCI Master Error Handling” on page 7-7 for additional details.

Reset Value x’0000 0000’

Address CPC0_PCIBAR + ’000F 9800’

Access Type Read/Write


Reserved
PCI Bus Address Parity Detected
G/P_SERR Detected
Invalid Memory Address
Reserved
Memory Error
PCI Bus Time-out on IRDY
Reserved
Arbitration Level
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**Reserved**
2	**PCI Bus Address Parity Detected** 0: No Error 1: PCI Bridge detected address parity error
3	**SERR Detected** 0: No Error 1: PCI Bridge detected G/P_SERR during transaction
4	**Invalid Memory Address** 0: No Error 1: PCI access occurred to invalid system memory address
5 - 7	**Reserved**
8	**Memory Error** 0: No Error 1: Double bit ECC error occurred during memory access
9	**Bus Time-out** 0: No Error 1: PCI Bridge detected bus time-out; no IRDY detected (see “PCICx_DISCNT” on page 9-66)
10	**Reserved**
11 - 15	**Arbitration Level.** Encoded arbitration level of PCI device when error occurred
16 - 31	**Reserved**

PCILx_CTRLW

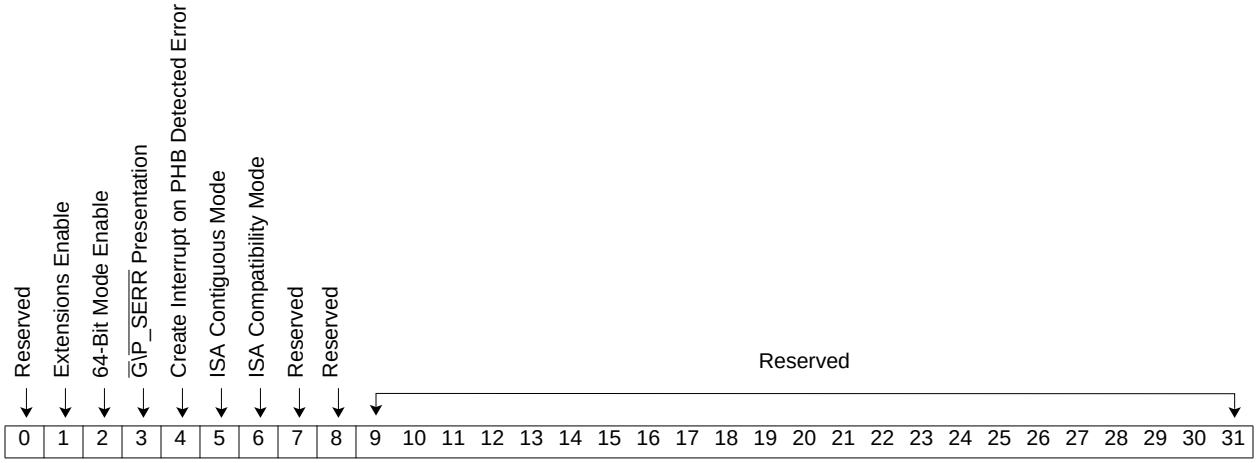
PHB Configuration Register

This register is primarily used by software to program the CPC710 for a particular address translation mode.

Reset Value x'0200 0000'

Address CPC0_PCIBAR + x'000F 7FD0'

Access Type Read/Write


Reserved
Extensions Enable
64-Bit Mode Enable
G\P_SERR Presentation
Create Interrupt on PHB Detected Error
ISA Contiguous Mode
ISA Compatibility Mode
Reserved
Reserved
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Reserved**
1	**Extensions Enable.** Read Only. 0: The CPC710 does not perform extended error recovery
2	**64-Bit Mode Enable.** Read Only. 0: The CPC710 does not support 64-bit addresses
3	**SERR Presentation.** Read Only. 0: PCI Bridge always generates Machine Check if $\overline{G}/\overline{P\_SERR}$ driven active
4	**Create Interrupt On PHB Detected Error.** Read Only. 0: PCI Bridge always generates Machine Check for PHB detected error
5	**ISA Contiguous Mode** This bit programs how the CPC710 translates the first 8 MB of PCI I/O space. See Figure 2-4 on page 2-5 for additional details. 0: ISA space is contiguous 1: ISA space is not-contiguous
6	**ISA Compatibility Mode.** Read Only. 1: The CPC710 contains an external pin for this function (P_ISA_MASTER).
7	**Reserved**
8	**Reserved**
9-31	**Reserved**

This register exists on PCI32 and PCI64. It enables the deadlock avoidance circuit in the CPC710 to manage and avoid deadlock situation that could result from concurrent access to/from one PCI bus.

The deadlock avoidance circuit is active when:

- At least one of the bits 0, 1, or 16 of the CPC0_ DLKCTRL register is set to 1

and

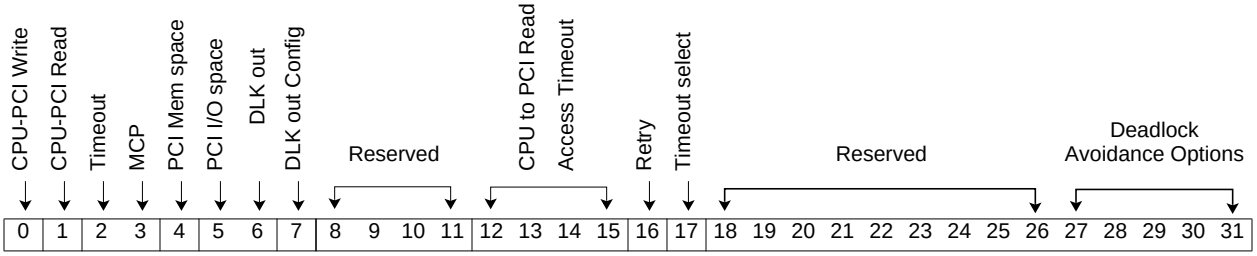
- CPC0_PGCHP register bit 24 is set to 0.

This inhibits the Retry signal on the PowerPC bus, and so avoids locking the bus.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8240'

Access Type Read/Write


CPU-PCI Write
CPU-PCI Read
Timeout
MCP
PCI Mem space
PCI I/O space
DLK out
DLK out Config
Reserved
CPU to PCI Read Access Timeout
Retry
Timeout select
Reserved
Deadlock Avoidance Options
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	Deadlock checking on CPU Write Access to PCI 0:No Deadlock checking 1:Management of Potential Deadlock
1	Deadlock checking on CPU Read Access to PCI 0:No Deadlock checking 1:Management of Potential Deadlock
2	Timeout Enable for CPU to PCI Read access 0:Timeout activation with the value defined in bits [12:15] for the event root defined in bit 17. 1:No timeout activation **(recommended value)**
3	MCP Activation 0:Machine Check (MCP) is active on TIMEOUT 1:No MCP activation
4	PCI MEMORY Space Selection 0:The PCILx_BPMDLK/PCILx_TPMDLK space is valid for PCI MEMORY Access 1:The PCILx_BPMDLK-PCILx_TPMDLK space is valid for PCI IO Access

PCILx_DLKCTRL

Deadlock Avoidance Control Register

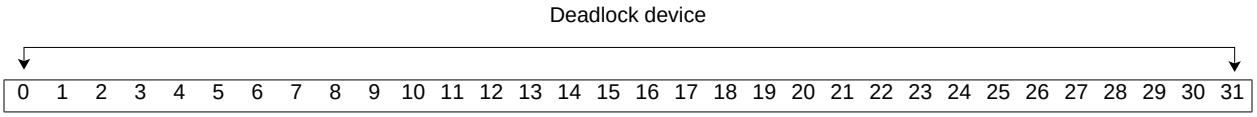
Bit(s)	Description
5	PCI IO Space Selection 0:The PCILx_BIODLK/PCILx_TIODLK space is valid for PCI IO Access 1:The PCILx_BIODLK/PCILx_TIODLK space is valid for PCI MEMORY Access
6	DLK_ Output Activation 0:This output is never activated 1:This output is activated for read or write access in the potential deadlock space selected
7	DLK_ Output Configuration 0:DLK_ output signal is used for PMDLK Space 1:DLK_ output signal is used for IODLK Space
8-11	Reserved - Must be left to 0
12-15	Timeout Value for CPU to PCI Read Access Defined as the number (2**n) of events for the timeout counter. Bit 17 is used to select the root event for the timeout count.
16	Deadlock management on PCI Configuration Access (Retry Activation) 0:No deadlock Space for the PCI Configuration space 1:The Deadlock space for the PCI Configuration space is defined by the PCILx_DLKDEV register (see *"PCILx_DLKDEV" on page 9-95*)
17	Event Root Select for timeout 0:Only retried CPU access to the PCI space which has activated the deadlock avoidance mechanism are used for the counter of timeout 1:All retried CPU access are used for the counter of timeout
18-27	**Reserved** - Read Only
27	DD2.0 ERRATA#9 correction 0: ERRATUM is corrected 1: ERRATUM is not corrected
28	DD2.0 ERRATA#8 correction 0: ERRATUM is corrected 1: ERRATUM is not corrected
29	ARTRY and Deadlock avoidance circuit improvement (recommended value is 0) 1: The ARTRY is generated for all access except on the access to the main memory when the checking of a Read is already in progress in the Deadlock avoidance logic circuit. 0: Modification not effective.
30	Processor ID and Deadlock avoidance circuit improvement (recommended value is 0) 0: Processor ID is taken into account in the DEADLOCK avoidance logic circuit 1: Modification not effective.
31	MEMREQ/MEMACK and Deadlock avoidance circuit improvement (recommended value is 0) 0: The Deadlock avoidance logic using MEMREQ/MEMACK and the Deadlock avoidance logic using DLK/NODLK are masked. 1: Modification not effective.

This register exists on PCI32 and PCI64.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8250'

Access Type Read/Write



Bit(s)	Description
0-31	**Deadlock Device** The Bit "n" corresponds to the PCI Device "n" During a PCI CONFIG access, the deadlock management is activated for the selected PCI devices. This operating mode is possible only if the bit 16 of PCILx_DLKCTRL is set to 1 (see *"PCILx_DLKDEV" on page 9-95)*. Example of coding: x00000000: No potential deadlock on the PCI configuration space. x00000002: device 2 is potentially in deadlock on the PCI CONFIG access. x00000004: device 3 is potentially in deadlock on the PCI CONFIG access. x00000008: device 4 is potentially in deadlock on the PCI CONFIG access. x00000006: devices 2 and 3 are potentially in deadlock on the PCI CONFIG access. etc.

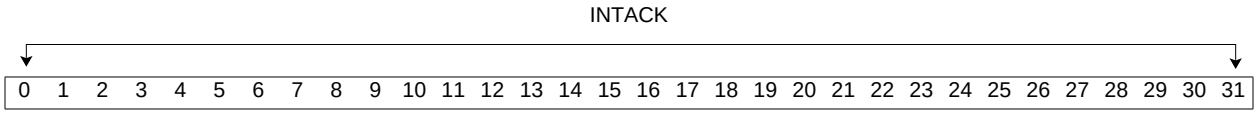
PCILx_INTACK

Interrupt Acknowledge Cycle

A read to the INTACK register generates an Interrupt Acknowledge Cycle on the PCI bus. An Interrupt Acknowledge Transaction has no addressing mechanism and is implicitly targeted to the interrupt controller in the system. The vector is returned by the interrupt controller when TRDY is asserted, on the PCI bus to the CPU waiting the "Read" Data.

Address CPC0_PCIBAR + x'000F 7700'

Access Type Read Only

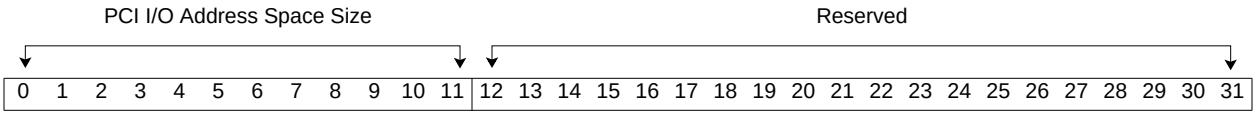


Bit(s)	Description
0 - 31	This register is a port through to the PCI bus. Writes to this register are ignored.

Reset Value x'FFF0 0000'

Address CPC0_PCIBAR + x'000F 7F60'

Access Type Read/Write



Bit(s)	Description
0-11	**PCI I/O Address Space Size** x'FFF' 1 MB x'FFE' 2 MB x'FFC' 4 MB x'FF8' 8 MB x'FF0' 16 MB x'FE0' 32 MB x'FC0' 64 MB x'F80' 128 MB x'F00' 256 MB x'E00' 512 MB x'C00' 1 GB x'800' 2 GB x'000' 4 GB
12-31	**Reserved**

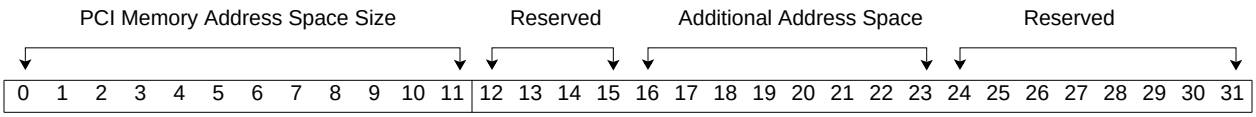
PCILx_MSIZE

PCI Memory Address Space Size

Reset Value x'FFF0 0000'

Address CPC0_PCIBAR + x'000F 7F40'

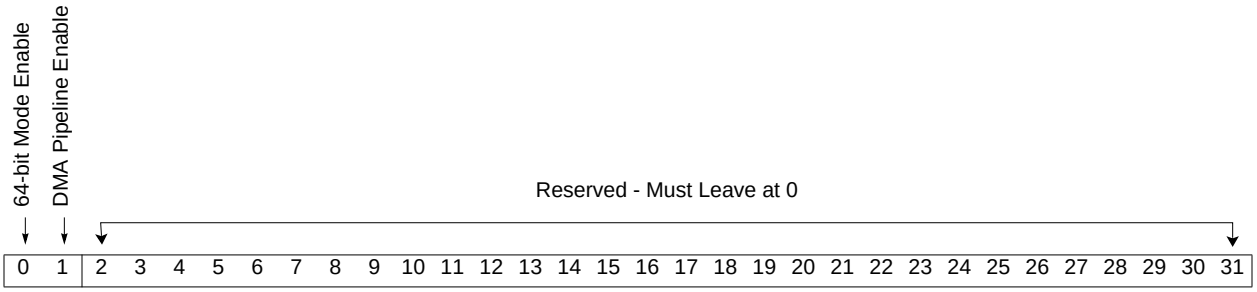
Access Type Read/Write


PCI Memory Address Space Size
Reserved
Additional Address Space
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0-11	**PCI Memory Address Space Size** x'FFF' 1 MB x'FFE' 2 MB x'FFC' 4 MB x'FF8' 8 MB x'FF0' 16 MB x'FE0' 32 MB x'FC0' 64 MB x'F80' 128 MB x'F00' 256 MB x'E00' 512 MB x'C00' 1 GB x'800' 2 GB x'000' 4 GB
12-15	**Reserved**
16-23	**PCI Memory additional Address Space Size (CHRP Compliance - Ignored in other modes)** x'FF' 16 MB x'FE' 32 MB x'FC' 64 MB x'F8' 128 MB x'F0' 256 MB x'E0' 512 MB x'C0' 1 GB x'80' no additional window x'00' no additional window
24-31	**Reserved**

This register contains two mode bits that are used for special modes of operation.

Reset Value	PCI32	x'4000 0000'
	PCI64	x'C000 0000'
Address	CPC0_PCIBAR + x'000F 6120'	
Access Type	Read/Write	



Bit(s)	Description
0	**64-bit Mode Enable (only for PCI 64)** **Operates only when the CPC710 is PCI64 bus master.** 0: Operates as a 32-bit bridge. $\overline{\text{G\_REQ64}}$ never activated by the CPC710. 1: Operates as a 64-bit bridge. $\overline{\text{G\_REQ64}}$ always activated for CPC710 initiated transfers.
1	**DMA Pipeline Enable** When pipelining is enabled, the CPC710 internal buffering is effectively doubled; two 32 byte buffers instead of one are used for data transfer. This improves DMA transfer performance. 0: DMA transfers are NOT pipelined 1: DMA transfers are pipelined
2 - 31	**Reserved**. Must be set to 0

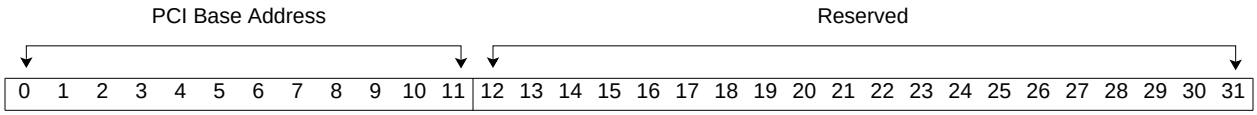
PCILx_PIBAR

PCI Base Address for I/O

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 7800'

Access Type Read/Write



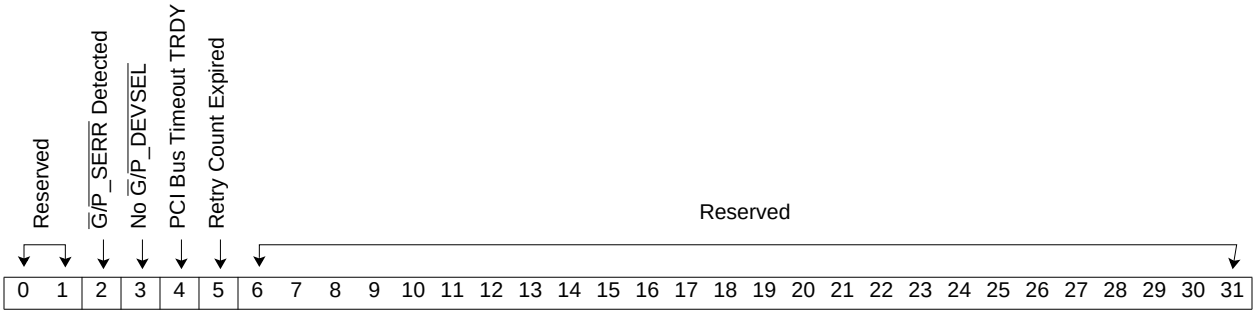
Bit(s)	Description
0 - 11	**PCI Base Address.** Contains the upper bits of the PCI base address that PCI I/O is mapped to. Note: Address must be aligned on boundary equal to size specified in PCI I/O Size register
12 - 31	**Reserved**

This register provides error status information for all transfers initiated by the CPU, a PCI master, or the other PCI Bridge logic. See “Error Handling for CPU-Initiated Transactions” on page 4-15 for additional details on this register.

Reset Value x’0000 0000’

Address CPC0_PCIBAR + x’000F 9810’

Access Type Read/Write


Reserved
G/P_SERR Detected
No G/P_DEVSEL
PCI Bus Timeout TRDY
Retry Count Expired
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**Reserved**
2	**SERR Detected** 0: No Error 1: PCI Bridge detected G/P_SERR active during master operation
3	**No Devsel** 0: No Error 1: PCI Bridge did not receive G/P_DEVSEL during master operation
4	**PCI Bus Timeout** 0: No Error 1: PCI Bridge detected bus time-out; no G/P_TRDY detected
5	**Retry Count Expired** 0: No Error 1: PCI Bridge detected bus time-out, too many retry’s (see “PCICx_RETRY” on page 9-73)
6 - 31	**Reserved**

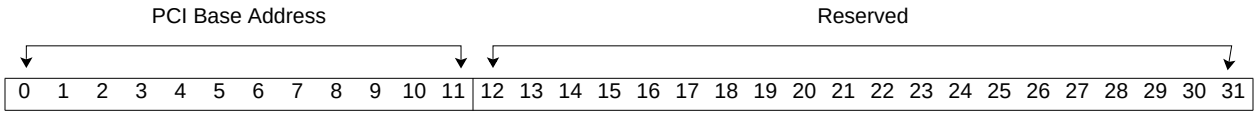
PCILx_PMBAR

PCI Base Address for Memory

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 7810'

Access Type Read/Write



Bit(s)	Description
0 - 11	**PCI Base Address.** Contains the upper bits of the PCI base address that PCI Memory is mapped to. **Note**: Address must be aligned on boundary equal to size specified in PCI Memory Size register.
12 - 31	**Reserved**

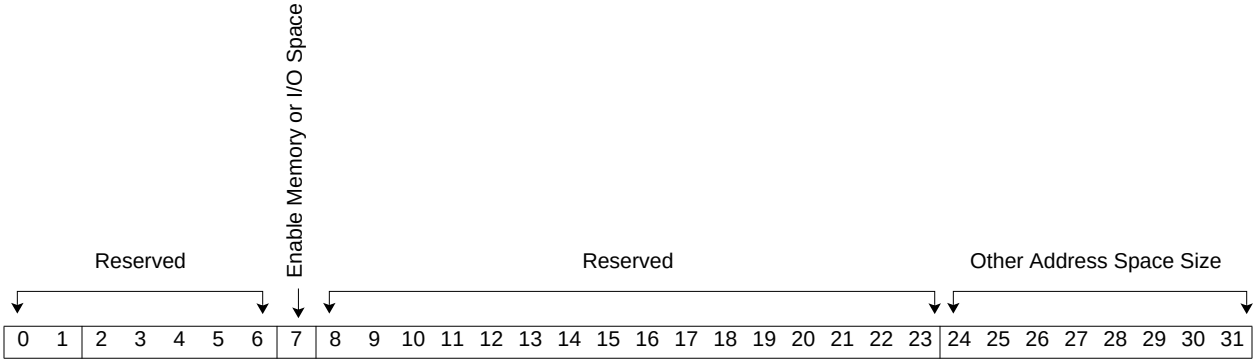
This is the same definition for 32-bit PCI and 64-bit PCI.

This register can be used to extend the memory space size up to 4G but **must be left to 0 if extended memory is not required**.
This functionality is available if bit 27 of CPC0_PGCHP register is set to 1 (page 9-26) and if FPHB mode or CHRP mode is selected in CPC0_PGCHP register (bits 0:3)

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8110'

Access Type Read/Write


Enable Memory or I/O Space
Reserved
Reserved
Other Address Space Size
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 6	**Reserved**
7	**Enable Memory or IO Space** 0: Memory Space 1: IO Space
8 - 23	**Reserved**
24 - 31	**Additional Address Space Size** x'FF' 16 MB x'FE' 32 MB x'FC' 64 MB x'F8' 128 MB x'F0' 256 MB x'E0' 512 MB x'C0' 1 GB x'80' 2 GB x'00' Access disabled

PCILx_PR

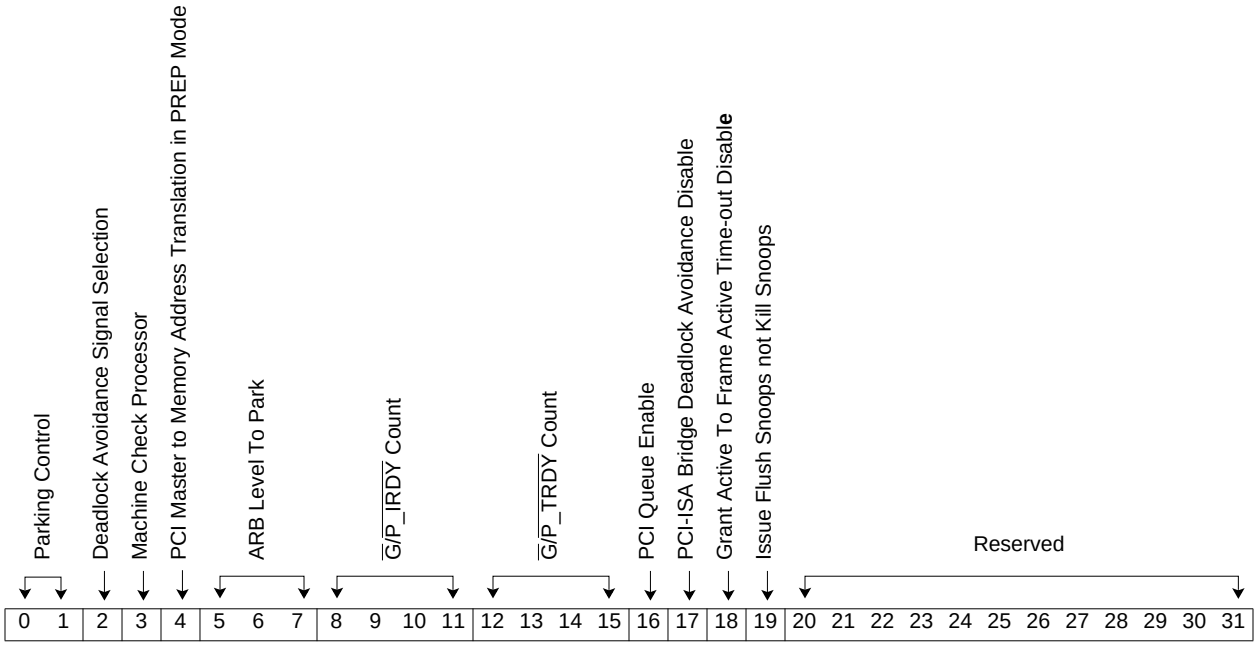
Personalization Register

This register provides additional programmability of the PCI Bridge logic.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 7F20'

Access Type Read/Write


Parking Control
Deadlock Avoidance Signal Selection
Machine Check Processor
PCI Master to Memory Address Translation in PREP Mode
ARB Level To Park
G/P_IRDY Count
G/P_TRDY Count
PCI Queue Enable
PCI-ISA Bridge Deadlock Avoidance Disable
Grant Active To Frame Active Time-out Disable
Issue Flush Snoops not Kill Snoops
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 1	**Parking Control** 0x: Bus is parked on CPC710 10: MRU algorithm for parking 11: Park specified ARB level below in bits 5-7
2	**Deadlock Avoidance Signal Selection (PCI32 Bridge Only)** To avoid deadlocks with PCI-ISA bridges on the PCI 32-bit bridge, the bridge must indicate to the CPC710 that a PCI access is about to occur before the P_GNT signal is activated. Any posted PCI 32-bit bus transfers must be flushed prior to activating the P_GNT signal and any accesses to the PCI 32-bit bus must be disabled after the GNT is given and continue disabled until the PCI access is complete and the P_GNT signal is removed. The PCI-ISA bridge must not grant the secondary ISA bridge until the CPC710 has activated the P_GNT signal. This bit should always be set to 1. The CPC710 provides two input signals for this purpose that are selectable with this bit. 0: Selects the P_REQ[5] signal. P_GNT[5] indicates buffers flushed and any PCI. Transfers will be disabled on the 60x bus until the P_REQ[5] signal is deactivated. 1: Selects the P_MEMREQ signal. P_MEMACK indicates buffers flushed and any PCI. Transfers will be disabled on the 60x bus until the P_MEMREQ signal is deactivated
3	**Machine Check Processor.** If an error is detected as a target during a PCI access operation, the CPC710 generates a Machine Check to the processor specified by the value of this register. 0: PCI bridge logic machine checks processor 0 1: PCI bridge logic machine checks processor 1

Bit(s)	Description
4	**PCI Master to Memory Address Translation in PREP Mode** 0:PCI Master addresses are always translated before being presented to system memory (see "CPC0_PCIBAR" on page 9-23) 1 PCI Master addresses are NOT translated and sent directly to system memory
5 - 7	**ARB Level To Park.** Contains the encoded arbitration level to park when bus is idle: level 000 is for agent **0**, level 001 for agent **1** and so on.
8 - 11	**IRDY Count.** Contains the number of PCI clocks times 8 that the CPC710 waits before detecting a time-out condition. A value of zero disables the time-out check.
12 - 15	**TRDY Count.** Contains the number of PCI clocks times 8 that the CPC710 waits before detecting a time-out condition. A value of zero disables the time-out check.
16	**PCI Queue Enable (CPU to PCI Access)** 0: PCI logic does not queue requests 1: PCI logic queues up to two operations
17	**PCI-ISA Bridge Deadlock Avoidance Disable** 0: PCI-ISA Bridge is present in the system. Therefore, the CPC710 will: 1) NOT deactivate P_GNT[5] even if other REQs become active (other REQs internally gated) 2) not activate P_GNT[5] until 60x bus has flushed all posted PCI 32-bit bus transfers. 1: PCI-ISA Bridge is NOT present in system, so CPC710 treats the P_REQ[5] signal like any other PCI bus REQ signal.
18	**Grant Active To Frame Active Time-out Disable** 0: If the CPC710 grants the PCI bus to a PCI master and other REQs are outstanding, the PCI master must activate the FRAME signal within 20 cycles or the CPC710 will deactivate its GNT signal. 1: Once CPC710 has granted the bus to a PCI device, it waits until it sees FRAME active from that device before deactivating its grant signal. Note: The 20 cycle count is not guaranteed. The timer runs continuously and therefore the CPC710 could remove the grant at any time.
19	**Issue Flush Snoops Instead Of Kill Snoops** 0: PCI bridge requests the 60x logic to perform Kill snoops on 60x bus for PCI to memory access as normal. 1: PCI bridge substitutes Flush snoops instead of Kill snoops to the 60x logic. This is to avoid a 604 coherency problem that exists for Kill snoop operations.
20-31	**Reserved**

PCILx_PSEA

PCI Slave Error Address Register

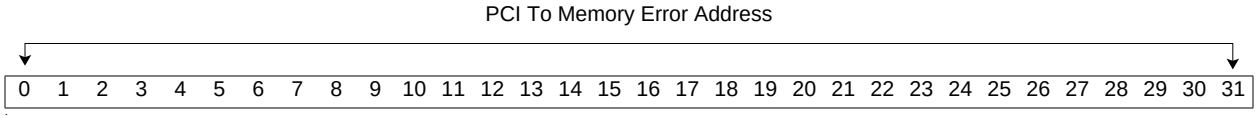
This register is used to log the PCI address when an error occurs during Device PCI slave transfer. See “PCI Master Error Handling” on page 7-7 for additional details.

This register is reset to zero after a POWERGOOD or when one of the bit RSTR[2] of “CPC0_RSTR” on page 9-32 for PCI32 or bit RSTR[3] for PCI 64 is forced to zero or from a software reset as described in “CPC0_SPOR” on page 9-41.

Reset Value x’0000 0000’

Address CPC0_PCIBAR + x’000F 6110’

Access Type Read Only



Bit(s)	Description
0 - 31	**PCI To Memory Error Address.** Contains the address present on the PCI bus when an error occurs during a PCI transfer.

This register exists on PCI32 and PCI64 and it controls the PCI to Memory Long Burst Read access.

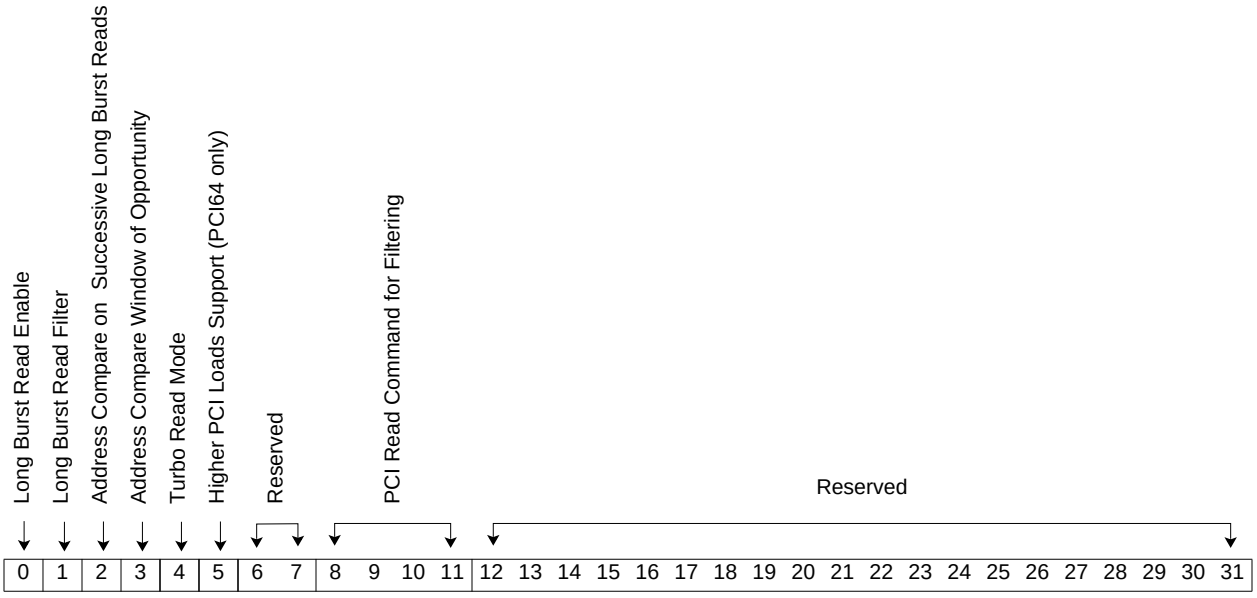
This register permits to improve performance for an I/O device master on the PCI bus when it is reading to the system memory through the CPC710.

The normal mode of burst transfer from a PCI Master to the Memory is 32 Bytes. The setting of bit 0 permit to have Long Burst of up to 4K Bytes with no Disconnect RETRY during the Burst.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8260'

Access Type Read/Write


Long Burst Read Enable
Long Burst Read Filter
Address Compare on Successive Long Burst Reads
Address Compare Window of Opportunity
Turbo Read Mode
Higher PCI Loads Support (PCI64 only)
Reserved
PCI Read Command for Filtering
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Long Burst READ Enable** 0 : Maximum burst size of 32 Bytes (**in this case all following bits must be left to 0**) 1 : Maximum burst size of 4 KBytes (No Disconnect PCICx_RETRY after 32 Bytes)
1	**Filtering of the type of READ converted in Long Burst Mode** (available only if bit 0 is set to 1) 0 : All access are in Long Burst mode 1 : Only the access with the PCI Command defined in bits [8:11] are in Long Burst mode
2	**Address Comparison On Successive Long Burst READ** (available only if bit 0 is set to 1) 0 : No Address Comparison made 1 : Latency for the first Data is reduced if the address has been already snooped in the previous Long Burst access. This is true if the new access is begun in the time window defined as below

PCILx_PSRCR

PCI Slave Read Control Register

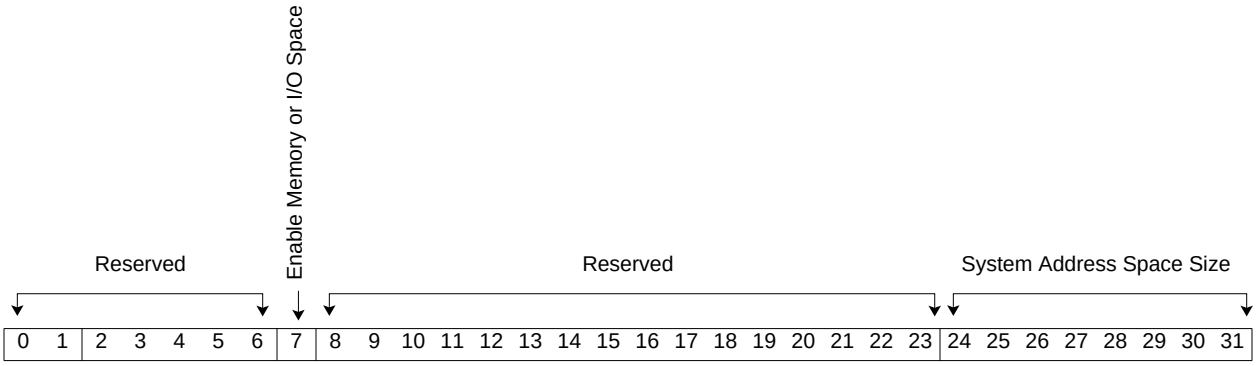
Bit(s)	Description
3	**Window opportunity for Address Comparison** The anticipation mode is possible only if there is at least a number of PCI clock cycles betweeen the end of one access and the start of the next access 0 : 8 PCI Cycles 1 : 64 PCI Cycles
4	**Turbo Read mode** (Only for the PCI64 and available only if bit 0 is set to 1) If selected the CPC710 use two additional 32 bytes buffers to increase memory bandwidth 0 : Disable 1 : Enable
5	**Higher PCI Loads Support** (Only for the PCI64) If selected the CPC710 responds with an additional wait state on TRDY# to avoid internal critical timing budget 0 : Disable 1 : Enable
6-7	**Reserved** - Read Only
8-11	**PCI READ Command C/BE# for Filtering option** (if enabled with bit 1) See C/BE[3:0] in the PCI section for supported commands.
12-31	**Reserved** - Read Only

This is the same definition for 32-bit PCI and 64-bit PCI.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8100'

Access Type Read/Write


Enable Memory or I/O Space
Reserved
Reserved
System Address Space Size
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 6	**Reserved**
7	**Enable Memory Or IO Space** 0: Memory Space 1: IO Space
8 - 23	**Reserved**
24 - 31	**System Address Space Size** x'FF': 16 MB x'FE': 32 MB x'FC': 64 MB x'F8': 128 MB x'F0': 256 MB x'E0': 512 MB x'C0': 1 GB x'80': 2 GB x'00': Access disabled

PCILx_PSWCR

PCI Slave Write Control Register

This register exists on PCI32 and PCI64, and it controls the PCI to Memory Long burst Write access.

This register permits to improve performance for an I/O device master on the PCI when it is writing to the system memory through the CPC710.

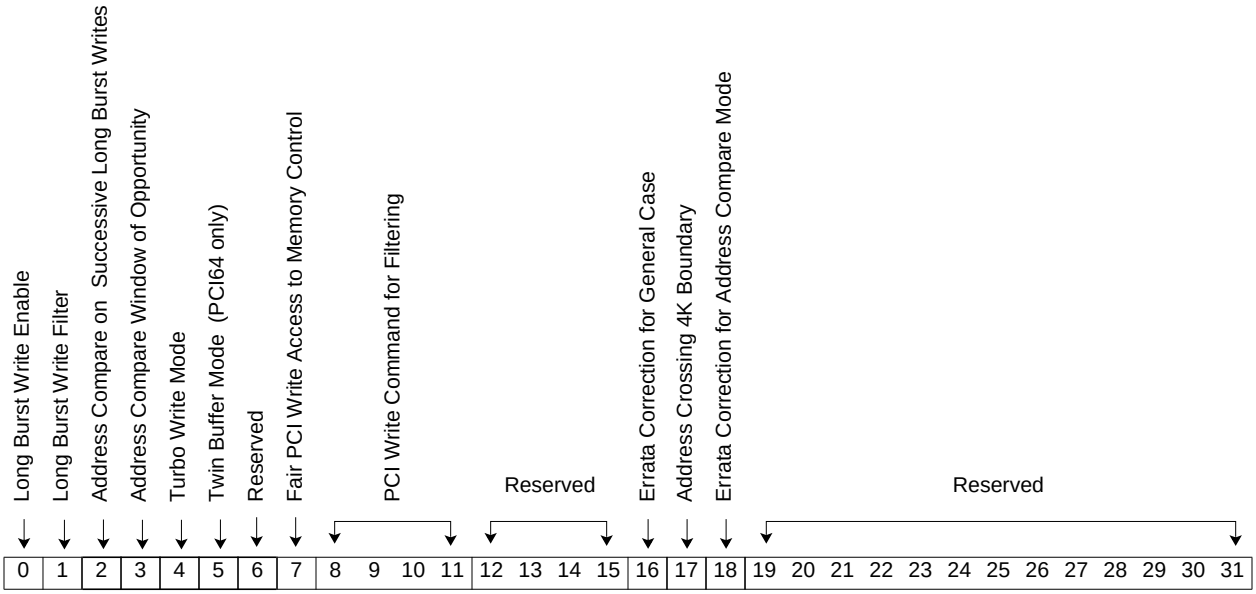
The normal mode of burst transfer from a PCI Master to the Memory is 32 Bytes.

The setting of bit 0 permit to have Long Burst of up to 4K Bytes with no Disconnect RETRY during the Burst.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8270'

Access Type Read/Write


Long Burst Write Enable
Long Burst Write Filter
Address Compare on Successive Long Burst Writes
Address Compare Window of Opportunity
Turbo Write Mode
Twin Buffer Mode (PCI64 only)
Reserved
Fair PCI Write Access to Memory Control
PCI Write Command for Filtering
Reserved
Errata Correction for General Case
Address Crossing 4K Boundary
Errata Correction for Address Compare Mode
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Long Burst WRITE Enable** 0 : Maximum burst size of 32 Bytes (**in this case all following bits must be left to 0**) 1 : Maximum burst size of 4 KBytes (No Disconnect PCICx_RETRY after 32 Bytes)
1	**Filtering of the type of WRITE converted in Long Burst Mode** (available only if bit 0 is set to 1) 0 : All access are in Long Burst mode 1 : Only the access with the PCI Command defined in bits [8:11] are in Long Burst mode
2	**Address Comparison On Successive Long Burst WRITE** (available only if bit 0 is set to 1) 0 : No Address Comparison made 1 : Latency for the first Data is reduced if the address has been already snooped in the previous Long Burst access. This is true if the new access is begun in the time window defined as below
3	**Window opportunity for Address Comparison** The anticipation mode is possible only if there is at least a number of PCI clock cycles between the end of one access and the start of the next access 0 : 8 PCI Cycles 1 : 64 PCI Cycles

Bit(s)	Description
4	**Turbo WRITE mode** (Only for the PCI64 and available only if bit 0 is set to 1) If selected the CPC710 make two consecutive anticipated snoop cycle and so reduce the number of wait states between data cache line 0 : Disable 1 : Enable
5	**Twin Buffer mode** (Only for the PCI64 and available **only if bit 0 and 4 are set to 1**) If selected the CPC710 use two additional 64 bytes buffers to increase memory bandwidth. Note: The PCILx_PSWCR[4:5] = 2b'01setting is not supported. 0 : Disable 1 : Enable
6	**Reserved** - Read Only
7	**Fair PCI Write Access to Memory Control** When enabled, re-arbitration for access to system memory will occur after a cache line (32 byte) transfer is completed. Re-arbitration will be among the PCI32, PCI64, DMA controller and the 60x bus interface. This allows for memory writes to take place during long burst read operations on PCI. 0: No PCI disconnect-Retry of a Long burst Read in progress 1: PCI Disconnect-Retry of a Long burst Read in progress instead of wait on PCI bus when CPC710's internal buffers are full.
8-11	**PCI WRITE Command C/BE# for Filtering option** (if enabled with bit 1) See C/BE[3:0] in the PCI section for supported Commands..
12-15	**Reserved** - Read Only
16	**DD2.0 ERRATA#7 correction for general case (must be left to 0 if bit 0 is not set to 1)** 0: ERRATUM is not corrected 1: ERRATUM is corrected
17	**Address Crossing 4K Boundary** 0 : 4K address crossing detection on PCI address (Anticipated Snoop cycle not used) 1 : 4K address crossing detection on Anticipated Snooping Address (save one cycle)
18	**DD2.0 ERRATA#7 correction for address comparison Mode (must be left to 0 if bit 2 is not set to 1)** 0: ERRATUM is not corrected 1: ERRATUM is corrected
19	**Reserved - Must be left to 0**
20-31	**Reserved** - Read Only

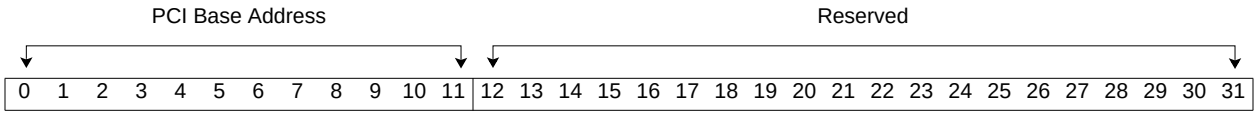
PCILx_SIBAR

System Base Address for PCI I/O

Reset Value x'8000 0000'

Address CPC0_PCIBAR + x'000F 7FC0'

Access Type Read/Write

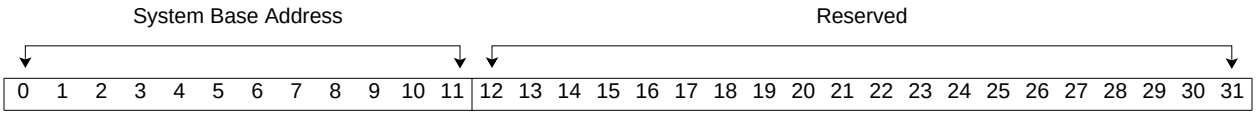


Bit(s)	Description
0 - 11	**System Base Address** This register contains the upper bits of the SYSTEM address that PCI I/O is mapped to. Note: Address must be aligned on boundary equal to size specified in PCI I/O Size register.
12 - 31	**Reserved**
Note: Address is decoded only if the Master Enable bit in the PCI Command Register is on.	

Reset Value x'A000 0000'

Address CPC0_PCIBAR + x'000F 7F80'

Access Type Read/Write



Bit(s)	Description
0 - 11	**System Base Address.** This register contains the upper bits of the SYSTEM address that PCI Memory is mapped to. Note: Address must be aligned on a boundary equal to the size specified in PCI Memory Size register.
12 - 31	**Reserved**
Note: Address is decoded only if the Master Enable bit in the PCI Command Register is on.	

PCILx_TIODLK

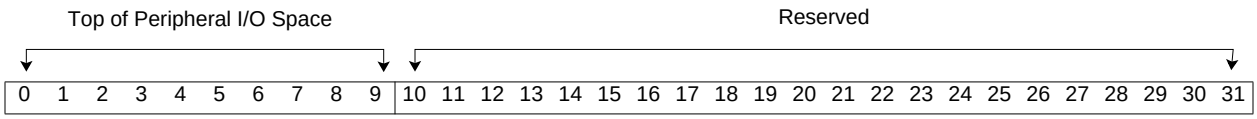
Top of Peripheral I/O space with Deadlock Avoidance

This register exists on PCI32 and PCI64.

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8230'

Access Type Read/Write

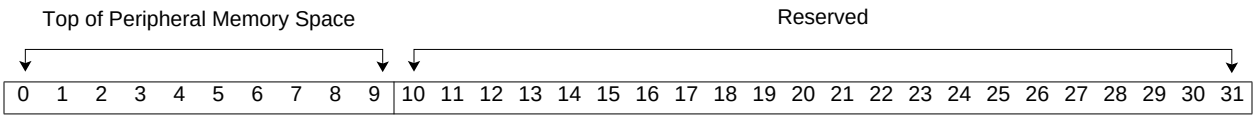


Bit(s)	Description
0 - 9	**Top of Peripheral I/O Space**. Contains the top address for the CPU to PCI IO access with potential deadlock
10 - 31	**Reserved**

Reset Value x'0000 0000'

Address CPC0_PCIBAR + x'000F 8210'

Access Type Read/Write



Bit(s)	Description
0 - 9	**Top of Peripheral Memory Space**. Contains the top address for the CPU to PCI MEMORY access with potential deadlock
10 - 31	**Reserved**

SDRAM0_MCCR

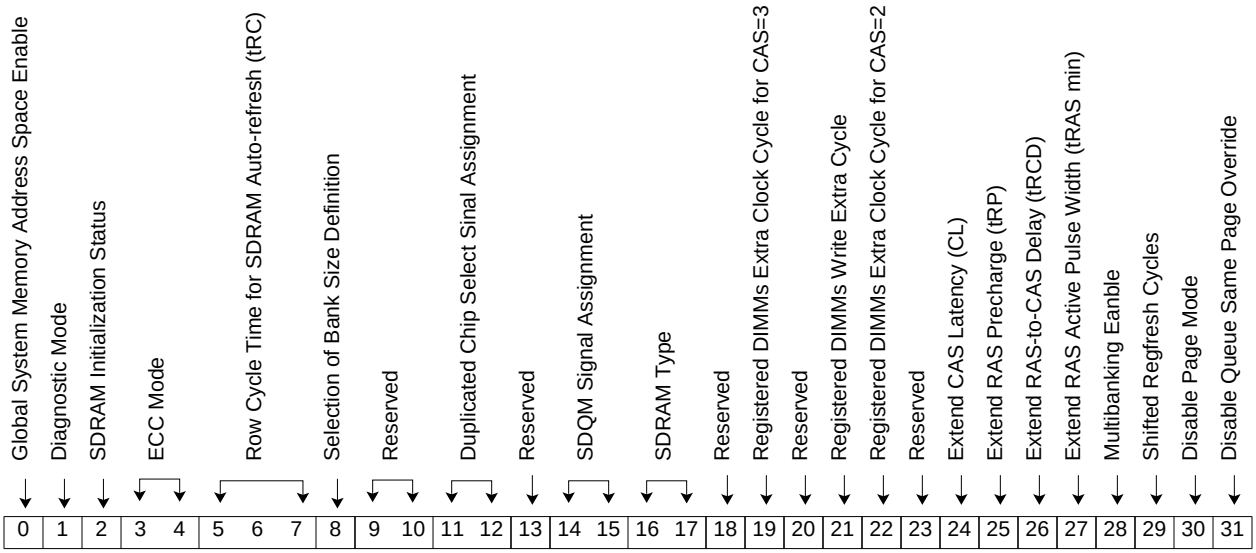
Memory Controller Control Register

This register provides the primary control for the memory controller logic.

Reset Value x'0000 0000'

Address x'FF00 1200'

Access Type Read/Write


Global System Memory Address Space Enable
Diagnostic Mode
SDRAM Initialization Status
ECC Mode
Row Cycle Time for SDRAM Auto-refresh (tRC)
Selection of Bank Size Definition
Reserved
Duplicated Chip Select Sinal Assignment
Reserved
SDQM Signal Assignment
SDRAM Type
Reserved
Registered DIMMs Extra Clock Cycle for CAS=3
Reserved
Registered DIMMs Write Extra Cycle
Registered DIMMs Extra Clock Cycle for CAS=2
Reserved
Extend CAS Latency (CL)
Extend RAS Precharge (tRP)
Extend RAS-to-CAS Delay (tRCD)
Extend RAS Active Pulse Width (tRAS min)
Multibanking Eanble
Shifted Regfresh Cycles
Disable Page Mode
Disable Queue Same Page Override
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	Global System Memory Address Space Enable 0:The CPC710 will not respond to addresses specified in Memory Configuration Extent Register (SDRAM0_MCERx) 1:System memory address space enabled.
1	Diagnostic Mode This bit is used to control presentation of double-bit ECC errors to the system. This bit is primarily intended for use in memory testing at power on time. Software can use this bit when testing memory and or ECC logic in order to avoid the hardware generating a machine check for double-bit ECC errors. The error however, is still logged into the SDRAM0_MEAR 0:Normal Mode: Multi-bit ECC error will generate Machine Check 1:Diagnostic Mode: Multi-bit ECC does NOT generate Machine Check; logged in SDRAM0_MEAR and SDRAM0_MESR
2	SDRAM Initialization Status (read-only) 0:SDRAM initialization is not completed. 1:SDRAM initialization is completed.

Bit(s)	Description
3 - 4	ECC Mode This field provides software with a means to control ECC generation and checking. b'01' is provided to allow software direct read/write access to the ECC byte that is associated with every doubleword of data stored in memory and also provide a mechanism to verify the memory controller's ECC generation and checking logic. In this mode, byte lane 0 data (MSB of a double word) is written to the ECC byte instead of the normal ECC code byte. Data byte 0 will be forced to all zeros. For reads, byte 0 will contain the byte stored in the ECC byte, not the data at byte 0. ECC checking is not enabled for reads in this mode. This mode also allows firmware write single-bit and multi-bit errors into memory to allow for ECC logic testing. 00:Normal generation and checking of ECC codes. The device will generate the normal ECC code when writing to memory and check ECC when reading. 01:ECC check disabled; Byte lane 0 routed to/from ECC check field. Data byte 0 forced to all zeros This mode is provided to allow software direct read/write access to the ECC byte that is associated with every double-word of data stored in memory and also provide a mechanism to verify the memory controller's ECC generation and checking logic. In this mode, byte lane 0 data (MSB of a double word) is written to the ECC byte instead of the normal ECC code byte. Data byte 0 will be forced to all zeros. For reads, byte 0 will contain the byte stored in the ECC byte, not the data at byte 0. ECC checking is not enabled for reads in this mode. This mode also allows firmware write single-bit and multi-bit errors into memory to allow for ECC logic testing. The device will still generate normal ECC codes when writing to memory. 10:ECC check disabled; Normal routing of data and normal ECC code generation The device will still generate normal ECC codes when writing to memory. 11:Reserved
5 - 7	Row Cycle Time for SDRAM Auto-refresh (t_{RC}) Allows to fit the delay between the Refresh Command and the next Activation. This delay has to be at least the tRCmin value specified in the SDRAM datasheet. 000:5 bus cycles 001:6 bus cycles 010:7 bus cycles 011:8 bus cycles 100:9 bus cycles 101:10 bus cycles 110:11 bus cycles 111:12 bus cycles
8	Selection of Bank Size Definition Select the encoding code for Size Bank defined in SDRAM0_MCER[16:25] bit field 1:Normal mode (4 MB to 1GB available) 0:Extended mode (4 MB to 4GB available)
9	**Reserved**. Must be Left to 0
10	**Reserved** (Must be set to 1 for SDRAM)
11	Chip Select duplicated Mode 1 1:SDCS_[0:3] signals are using SDCS_[4:7] outputs by Multiplexing
12	Chip Select duplicated Mode 2 1:SDCS_[0:3] signals are using SDCS_[8:11] outputs by Multiplexing
13	Reserved. Must be set to 0
14	First Multiplexing Control of SDRAM Signals SDQM 1: SDQM signals are using SDRAS1_, SDCAS1_ and WE1_ outputs by Multiplexing

SDRAM0_MCCR

Memory Controller Control Register

Bit(s)	Description
15	Second Multiplexing Control of SDRAM Signals SDQM 1: SDQM signals are using PCG_ARB outputs by Multiplexing
16-17	SDRAM Type 00: Standard modules SDRAM 01: Registred DIMMs SDRAM mode 2 (not available : for future use) 10: Registred DIMMs SDRAM mode 1 11: Reserved (unpredictable result)
18	**Reserved.** Must be left to 0
19	Registered DIMMs extra clock cycle for CAS=3 1:Following signals are shifted by one Clock cycle MUX_CLKEN1B_ MUX_CLKEN2B_ MUX_SEL_
20	**Reserved.** Must be set to 1
21	Registered DIMMs Write extra cycle: 1:Data to be written to the SDRAM Memory is maintained one more cycle
22	Registered DIMMs extra clock cycle for CAS=2 1:Following signals are internal shifted by one Clock cycle (SDCKE and SDCS signals must be external shifted by one Clock cycle): MUX_CLKENA2_ MUX_OEB_ SDRAS_0 SDRAS_1 SDCAS_0 SDCAS_1 WE_0 WE_1 MADDR0_ODD MADDR0_EVEN MADDR1-13 BS0 and BS1
23	**Reserved.** Must be left to 0
24	Extend CAS Latency (CL) 0:CAS Latency programmed to 2 cycles. 1:CAS Latency programmed to 3 cycles.
25	Extend RAS Precharge (tRP) 0:RAS Precharge programmed to 2 cycles. 1:RAS Precharge programmed to 3 cycles.

Bit(s)	Description
26	Extend RAS-to-CAS Delay (tRCD min) 0:RAS-to-CAS Delay programmed to 2 cycles. 1:RAS-to-CAS Delay programmed to 3 cycles.
27	Extend RAS Active Pulse Width (tRAS min) 0:Programmed to 5 cycles. 1:Programmed to 6 cycles.
28	Multibanking Enable 0:Multibanking desactived. 1:Multibanking is active.
29	Shifted Refresh Cycles In normal operation, the refresh of all the populated DIMMs is performed at the same time. If the 8 DIMMs are fully populated, this could produce a high current load (all SDCS_ activated at the same time). Setting bit 29 to 1, enables the controller to perform successive Refresh (only 2 SDCS_ activated at the same time); in return the refresh cycle last longer depending on the number of populated DIMMs. 0:All banks are refreshed in the same cycle. 1:Banks are refreshed one after one.
30	Disable Page Mode 0:Memory controller will perform fast page accesses for back to back operations if appropriate 1:Memory controller will perform fast page access only within a burst operation. It will NOT perform fast page accesses for back to back bursts even if they occur to the same $\overline{RAS}$ page.
31	Disable Queue Same Page Override 0:Memory queue ordering can be overridden if an operation is to the same page. To avoid hang conditions, leave this bit set to 0. See errata listing for more details. 1:Memory queue always processed in order received.

SDRAM0_MCER0:5

Memory Configuration Extent Registers 0:5

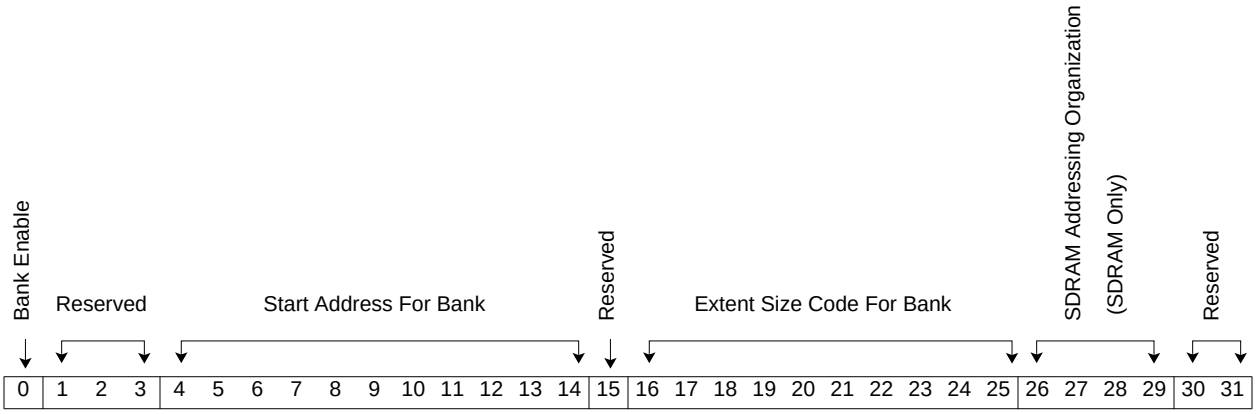
Each one of the six registers SDRAM0_MCER0 to SDRAM0_MCER5 defines one of up to six banks of memory (Bank 0 to 5) supported. All registers have the same definition, and each defines the size and location for the particular bank of memory.

Reset Value x'0000 0000'

Address

SDRAM0_MCER0	x'FF00 1300'
SDRAM0_MCER1	x'FF00 1310'
SDRAM0_MCER2	x'FF00 1320'
SDRAM0_MCER3	x'FF00 1330'
SDRAM0_MCER4	x'FF00 1340'
SDRAM0_MCER5	x'FF00 1350'

Access Type Read/Write


Bank Enable
Reserved
Start Address For Bank
Reserved
Extent Size Code For Bank
SDRAM Addressing Organization
(SDRAM Only)
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Bank Enable** 0: Bank is not present or register initialization is not complete 1: Bank is present and decoded by the CPC710
1 - 3	**Reserved**. R/W
4 - 14	**Start Address For Bank.** (Bits 0 - 10) - Defines the beginning address of this bank. Contains upper bits 0 - 10 of the 32-bit real address. Address restricted to a boundary equal to the size of the bank.
15	**Reserved**. R/W

Bit(s)	Description
16 - 25	**Size Code For Bank** The encoding code depends of the programmed bit 8 of the Memory Controller Control Register (SDRAM0_MCCR): **If SDRAM0_MCCR[8] = 1, the size code is defined as below:** x'3FF' Reserved x'3FB' Reserved x'3F3' 4 MB x'3E3' 8 MB x'3C3' 16 MB x'383' 32 MB x'303' 64 MB x'203' 128 MB x'003' 256 MB x'002' 512 MB x'000' 1 GB **If SDRAM0_MCCR[8] = 0, the size code is defined as below:** x'3FF' 4 MB x'3FE' 8 MB x'3FC' 16 MB x'3F8' 32 MB x'3F0' 64 MB x'3E0' 128 MB x'3C0' 256 MB x'380' 512 MB x'300' 1 GB x'200' 2 GB x'000' 4 GB

SDRAM0_MCER0:5

Memory Configuration Extent Registers 0:5

Bit(s)	Description
26 - 29	**SDRAM Addressing Organization** b’0001’: 11/ 8/2 (Row/Col/Bank select) Address lines b’0010’: 11/ 9/1 b’0011’: 11/10/1 b’0100’: 12/ 8/2 b’0101’: 12/10/2 b’0110’: 13/ 8/1 b’0111’: 13/ 8/2 b’1000’: 13/ 9/1 b’1001’: 13/10/1 b’1010’: 11/ 8/1 b’1011’: 12/ 8/1 b’1100’: 12/9/1 b’1101’: 14/9/2 b’1110’: 14/10/2 b’1111’: 14/11/2 b’0000’: All other supported organizations (see “SDRAM Subsystem Overview” on page 6-6) 12/9/2 13/9/2 13/10/2 13/11/2 13/12/2 14/12/2
30 - 31	**Reserved**

SDRAM0_MEAR

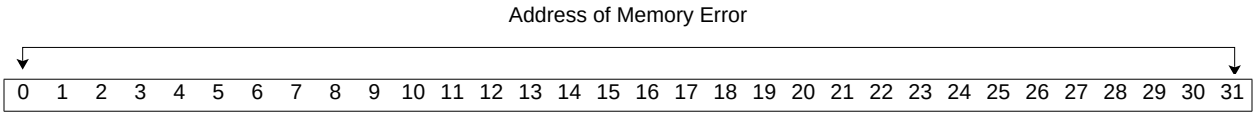
Memory Error Address Register

This register contains the address associated with the error logged in SDRAM0_MESR.

Reset Value x'0000 0000'

Address x'FF00 1230'

Access Type Read/Write



Bit(s)	Description
0 - 31	**Address of Memory Error**

SDRAM0_MESR

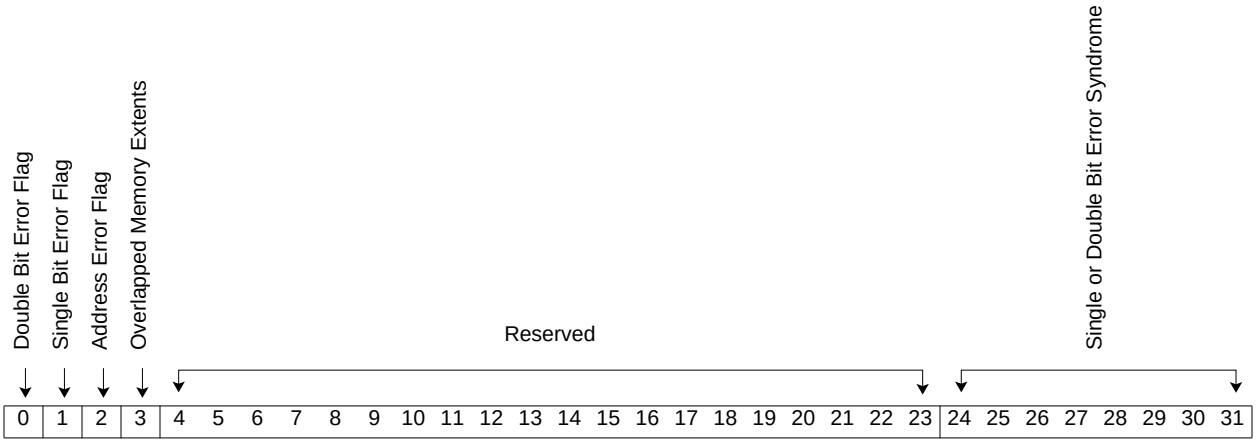
Memory Error Status Register

This register provides error status information on memory errors. In order to log additional errors, software must clear the register by writing zeros throughout.

Reset Value x'0000 0000'

Address x'FF00 1220'

Access Type Read/Write


Double Bit Error Flag
Single Bit Error Flag
Address Error Flag
Overlapped Memory Extents
Reserved
Single or Double Bit Error Syndrome
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0	**Double Bit Error Flag** 0: No Error 1: Double Bit ECC error occurred
1	**Single Bit Error Flag** 0: No Error 1: Single bit ECC error occurred
2	**Address Error Flag** 0: No Error 1: Address error occurred
3	**Overlapped Memory Extents** 0: No Error 1: An access occurred to an address that is mapped in two different memory configuration extents.
4 - 23	**Reserved**
24 - 31	**Single or Double Bit Error Syndrome.** Used to determine the failing DIMM

SDRAM0_MWPR

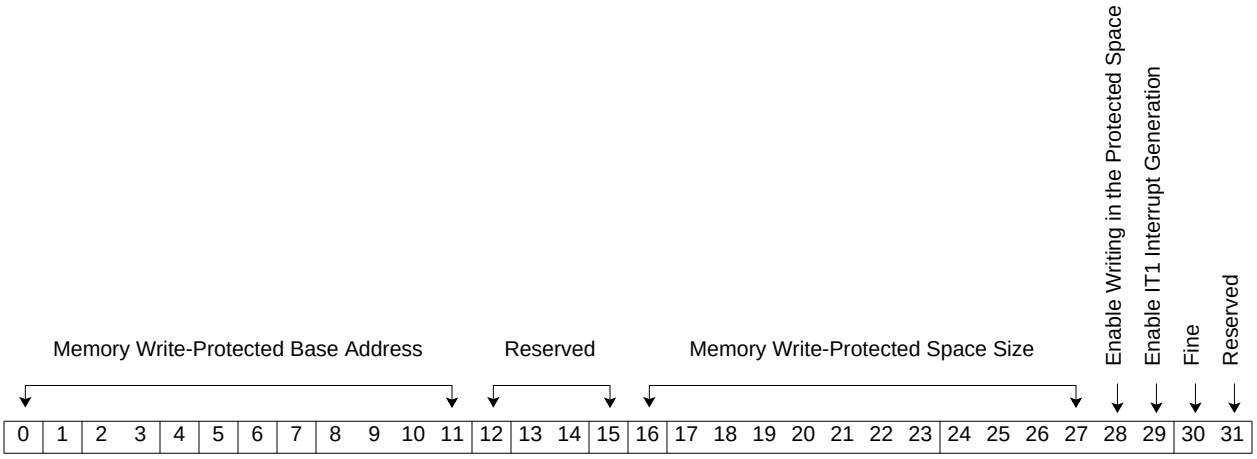
Memory Write Protection Register

SDRAM0_MWPR is used to protect write access to a selected memory space, from the PowerPC CPU or a master on a PCI bus.This feature is available with the use of the SDQM signals of the SDRAM.

The CPC710 cannot write to memory space addresses specified in this register.

To write to the protected memory space, the field MWPR[16:27] must be set to Zero.

Reset Value	x'0000 0000'
Address	x'FF00 1210'
Access Type	Read/Write


Memory Write-Protected Base Address
Reserved
Memory Write-Protected Space Size
Enable Writing in the Protected Space
Enable IT1 Interrupt Generation
Fine
Reserved
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0-11	**Memory Write-Protected Base Address** This register contains the upper bits of the Memory base address of the write-protected Space. Note: Address must be aligned on boundary equal to size specified in Memory Size register.
12-15	**Reserved**
16-27	**Memory Write Protected Space Size** x'FFF' 1 MB xF'FE' 2 MB x'FFC' 4 MB x'FF8' 8 MB xF'F0' 16 MB x'FE0' 32 MB x'FC0' 64 MB x'F80' 128 MB x'F00' 256 MB x'E00' 512 MB x'C00' 1 GB x'800' 2 GB x'000' No Write Protected Space
28	**Enable Writing in the protected space.** Inactive (=0) a write is possible, Active (=1) the write is not possible. Both cases differs on the memory side by activation of the SDRAM Data Mask (DQM_) signals. It is thus necessary to set the bit 11 of the SDRAM0_MCCR[11] = 1 to inhibit writing.

SDRAM0_MWPR

Memory Write Protection Register

Bit(s)	Description
29	**Enable IT1 Interrupt** generation when target address is in the Write-Protect zone. IT1 is active during 4 system cycles.
30	**Bit FINE** when =1 finer granularity is set (minimum = 4 KB) The result is unpredictable if bit CPC0_PGCHP[27]=1 0: If FINE bit is not active (reset = 0) the memory size is decoded on the 12 MSB bits of the memory address => minimum memory step size 1 MB. 1: If FINE bit is active (=1) the address have 8 more bits. The minimum memory step size is 4 KB with address on 20 bits. The 8 additional bit (LSB) of the memory Base address are in bits [24:31] of PCIC1_PPBAR or PCIL0_PPBAR (PCI interface register address 000F8150), The 8 additional bit (LSB) of the protected Space Size are in the Register PCILx_PPSIZE[24:31] (PCI interface register address 00F8110).
31	Reserved

Example: How to protect in write a memory zone of 8 KB from address 0x02024000 to 0x02025FFF

SDRAM0_MWPR[0:11] = 0x020

SDRAM0_MWPR[16:27] = 0xFFF

PCIL0_PPBAR[24:31] = 0x24

PCIL0_PPSIZE[24:31] = 0xFE

SDRAM0_MWPR[30] = 1

This register is user defined.

However it has been introduced in the PowerPC chip support to provide the memory DIMM presence detect pins for all four pairs of DIMM sockets. The CPC710 supports a maximum of four pairs, or eight DIMMs. The DIMM pairs must be of exactly the same type and therefore only one DIMM presence detect pin of each pair are read in from this register.

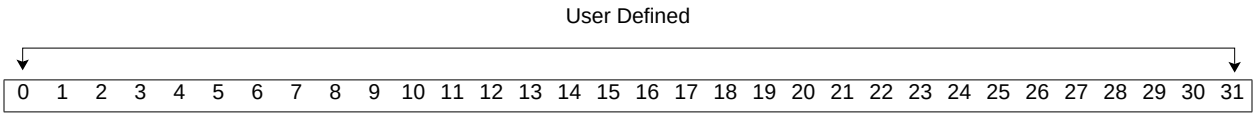
The read of this register results in the assertion of the $\overline{\text{PRES\_OE0}}$ signal and a Read cycle through the PCI32 A/D lines. That permits a read of the outside buffers containing the presence detect bits.

Bit 0 of this register correspond to bit 31 on the PCI A/D lines.

Reset Value x'0000 0000'

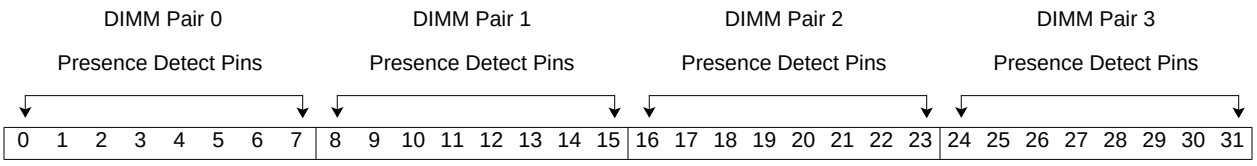
Address x'FF00 1400'

Access Type Read Only



Bit(s)	Description
0 - 31	**User-defined**

Example of usage:


DIMM Pair 0
Presence Detect Pins
DIMM Pair 1
Presence Detect Pins
DIMM Pair 2
Presence Detect Pins
DIMM Pair 3
Presence Detect Pins
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Bit(s)	Description
0 - 7	**DIMM Pair 0 Presence Detect Pins: PD1-PD8**
8 - 15	**DIMM Pair 1 Presence Detect Pins: PD1-PD8**
16 - 23	**DIMM Pair 2 Presence Detect Pins: PD1-PD8**
24 - 31	**DIMM Pair 3 Presence Detect Pins: PD1-PD8**

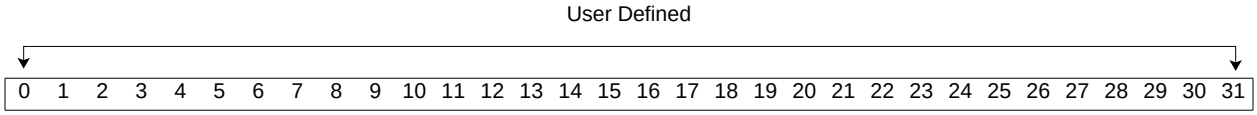
Please see "SDRAM0_MCERx Register Initialization" on page 6-15 for device supported values.

SDRAM0_SIOR1

System I/O Register 1

Address x‘FF00 1420’

Access Type Read Only



Bit(s)	Description
0 - 31	**User-defined**

The read of this register results in the assertion of the $\overline{\text{PRES\_OE1}}$ signal and a Read cycle through the PCI32 A/D lines. That permits a read of the outside buffers containing the presence detect bits.

Bit 0 of this register correspond to bit 31 on the PCI A/D lines.

Chapter 10. Timing Diagrams

10.1 CPU to Memory Transactions

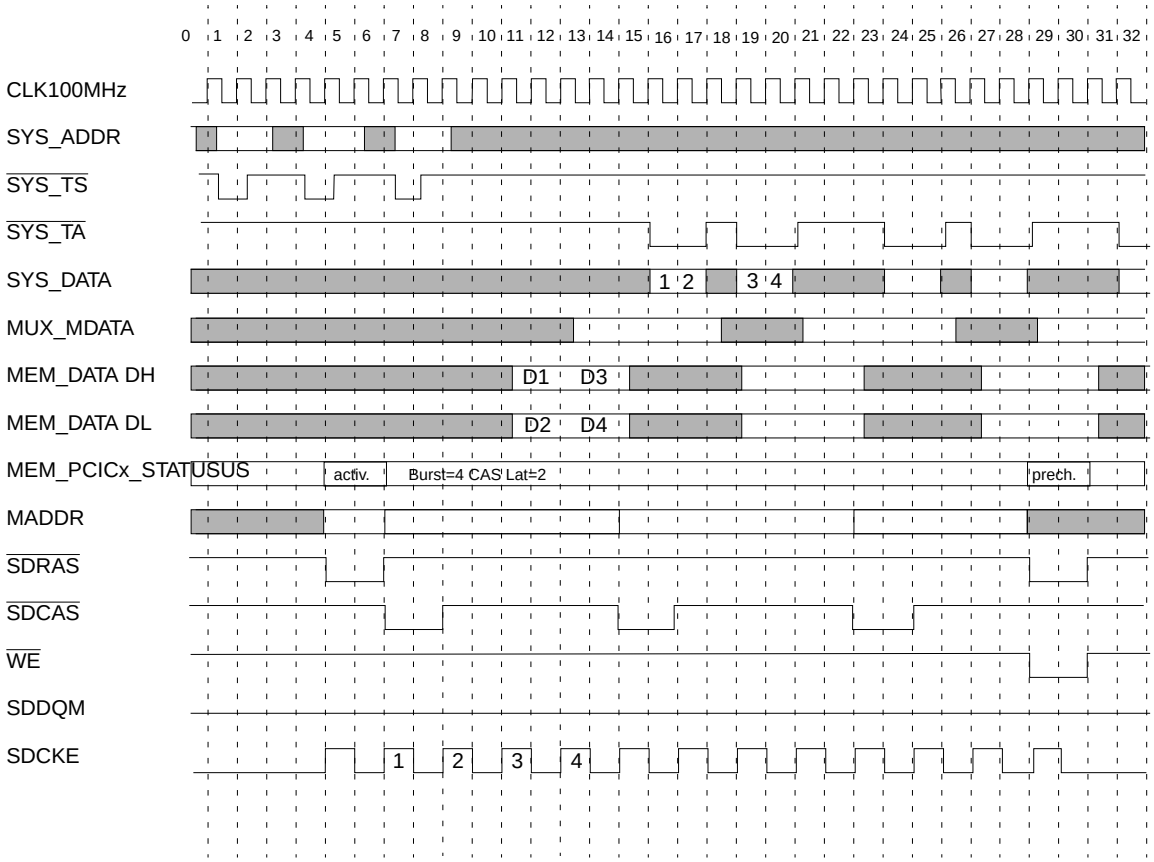


Figure 10-1. Read Page Hit from PowerPC CPU to SDRAM

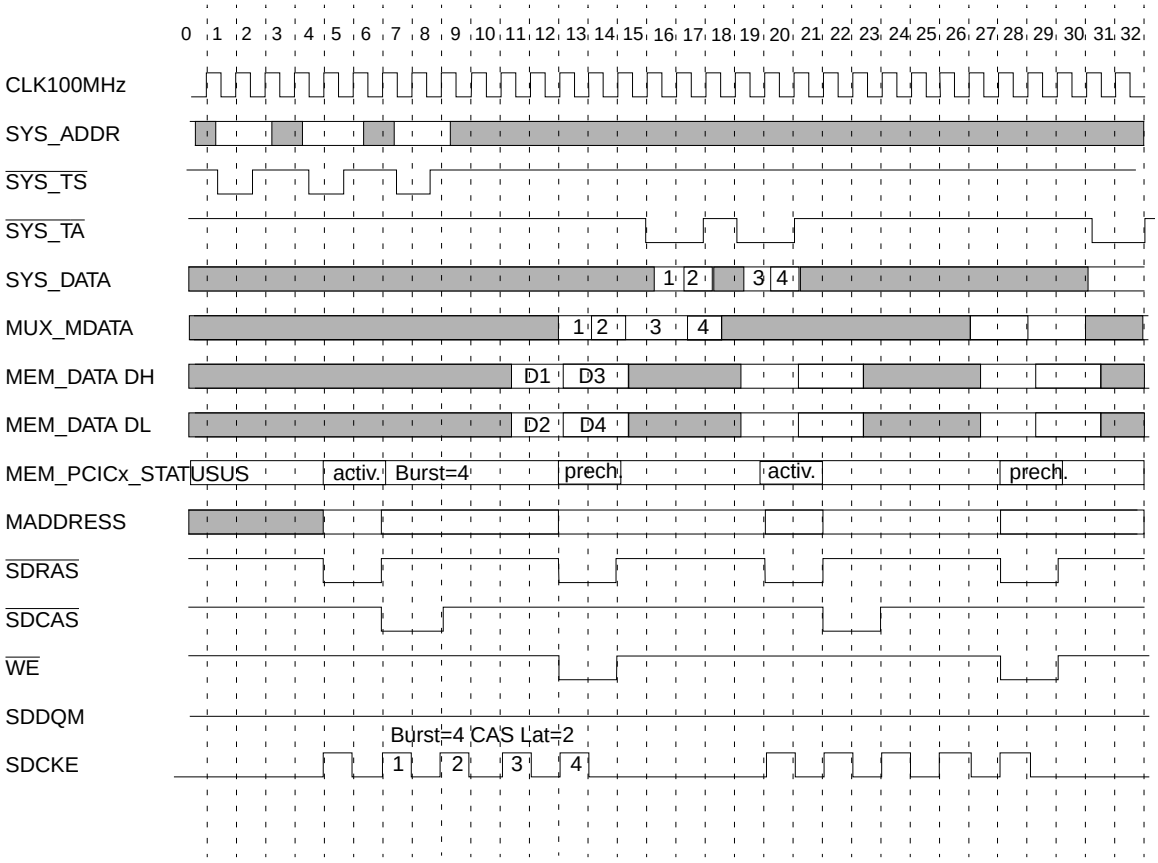


Figure 10-2. Read Page Miss from PowerPC CPU to SDRAM

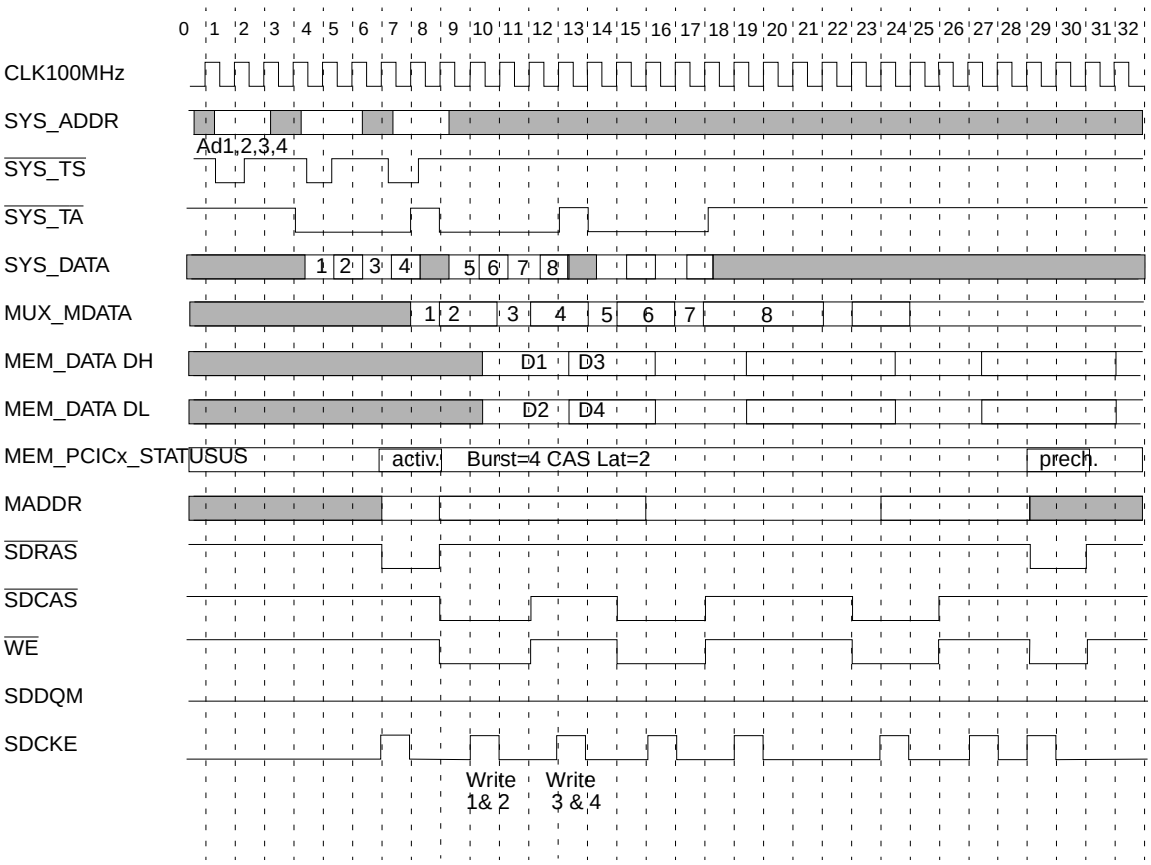


Figure 10-3. Write Burst Page Hit from PowerPC CPU to SDRAM

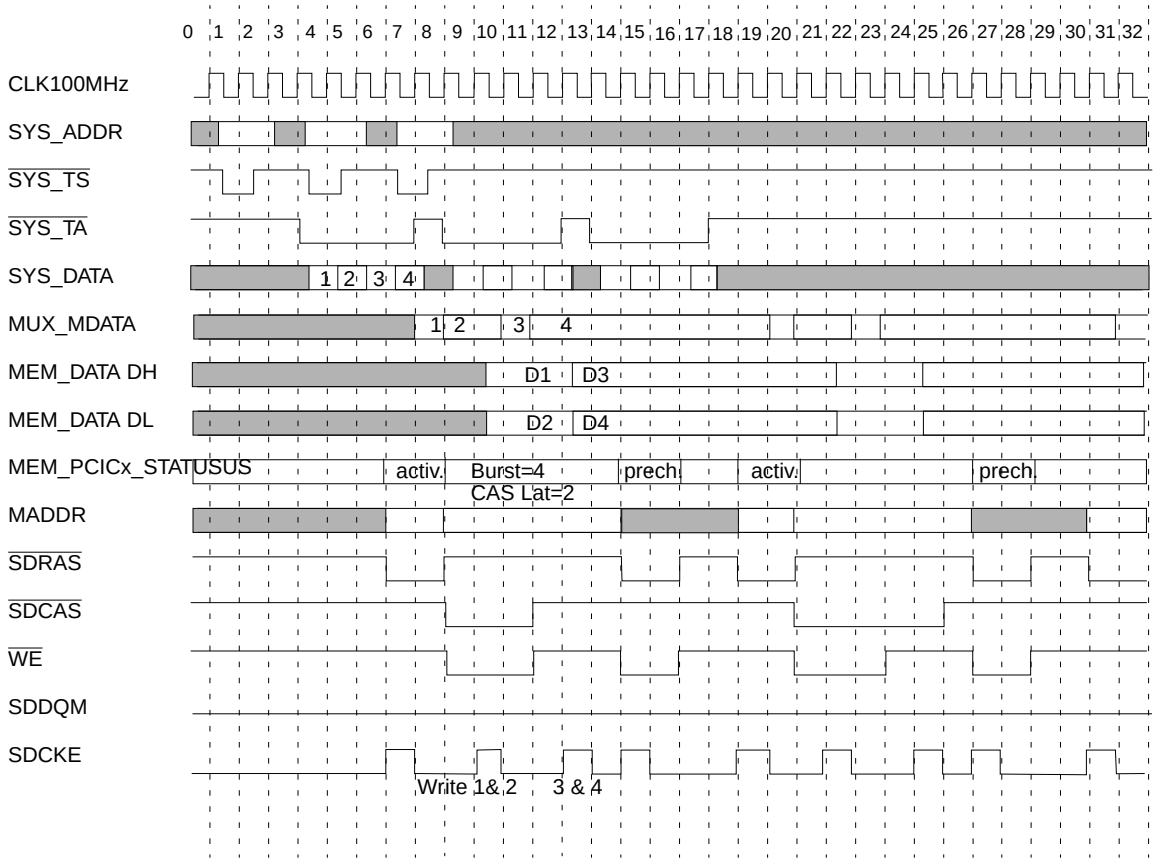


Figure 10-4. Write Burst Page Miss from PowerPC CPU to SDRAM

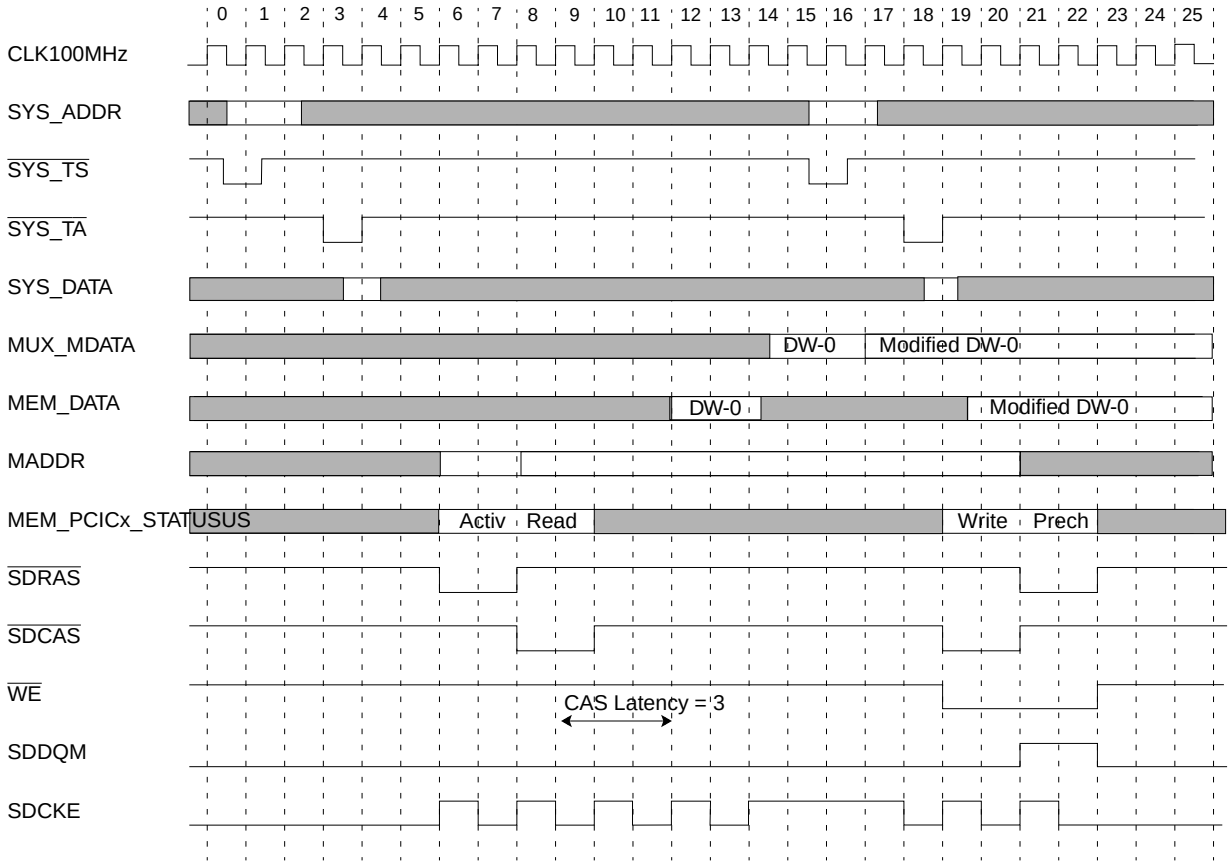


Figure 10-5. Write One Byte to Memory from CPU: Read Modify Write

10.2 CPU Access to the Boot ROM

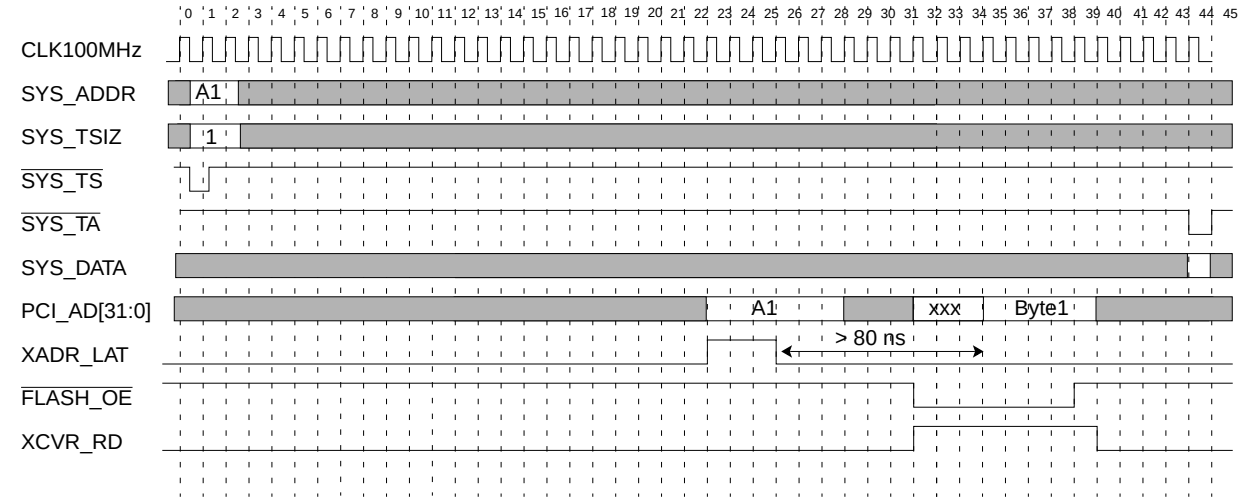


Figure 10-6. Read of One Byte from the Boot ROM

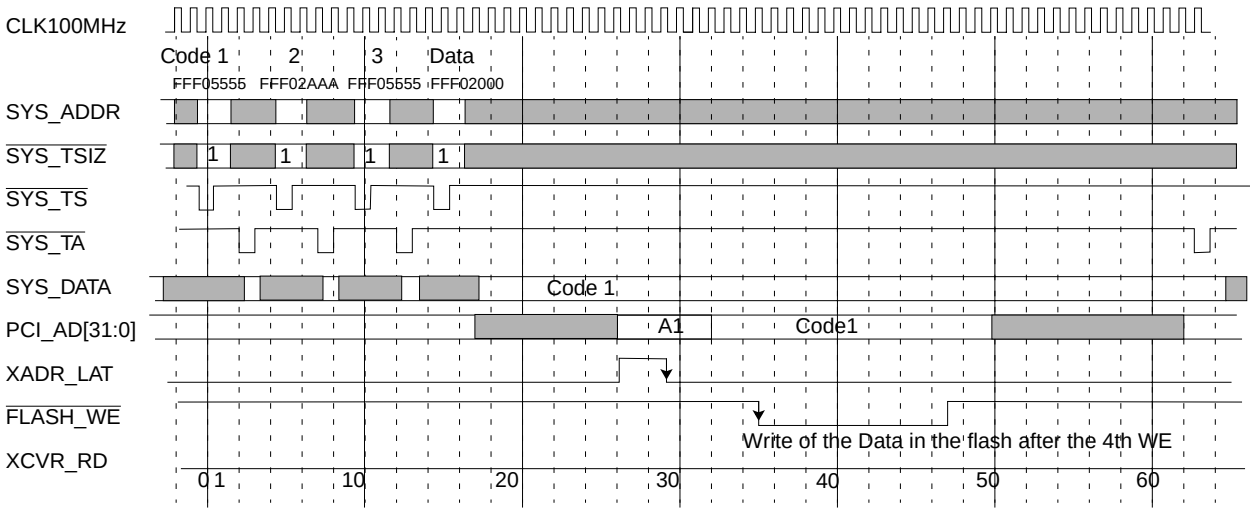
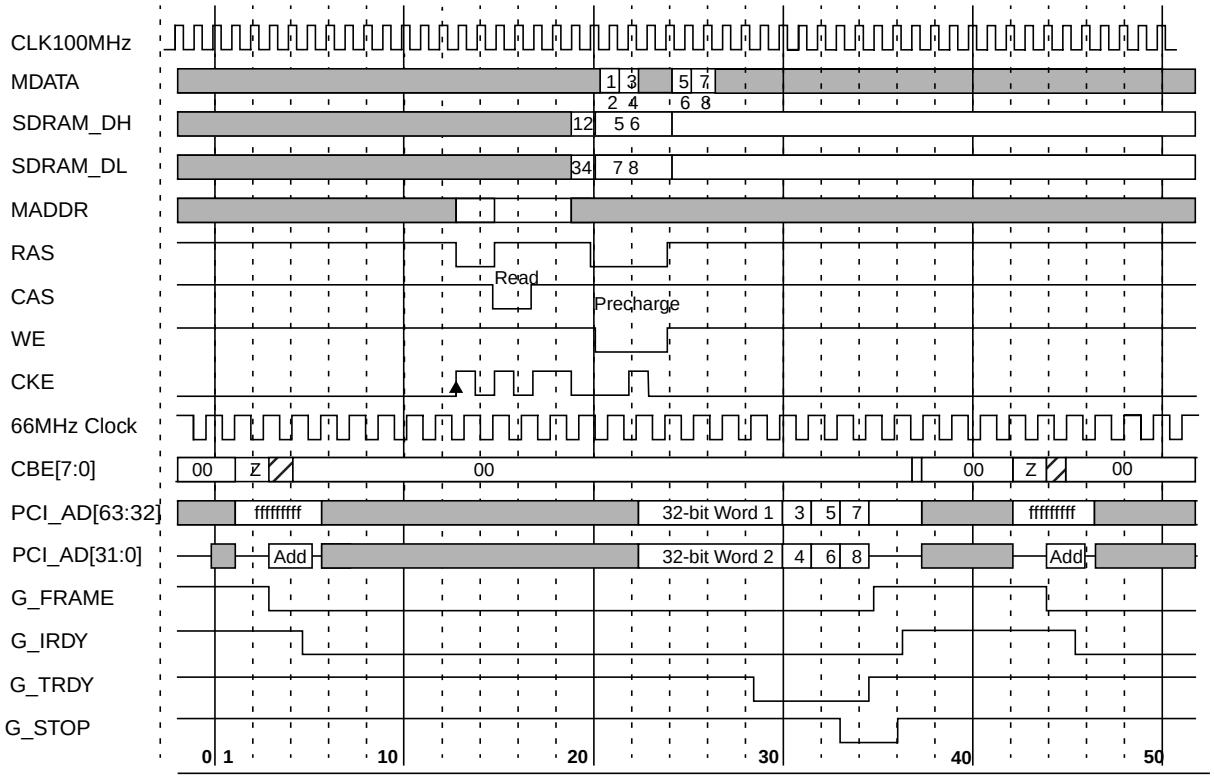


Figure 10-7. Write of One Byte to the Boot Flash

10.3 PCI64 External Master Accessing SDRAM Memory



Note: 1,2,3 .. refers to each 32-bit word.

Figure 10-8. Read 32 Bytes from SDRAM by a PCI Master on a 66MHz PCI64 bus:

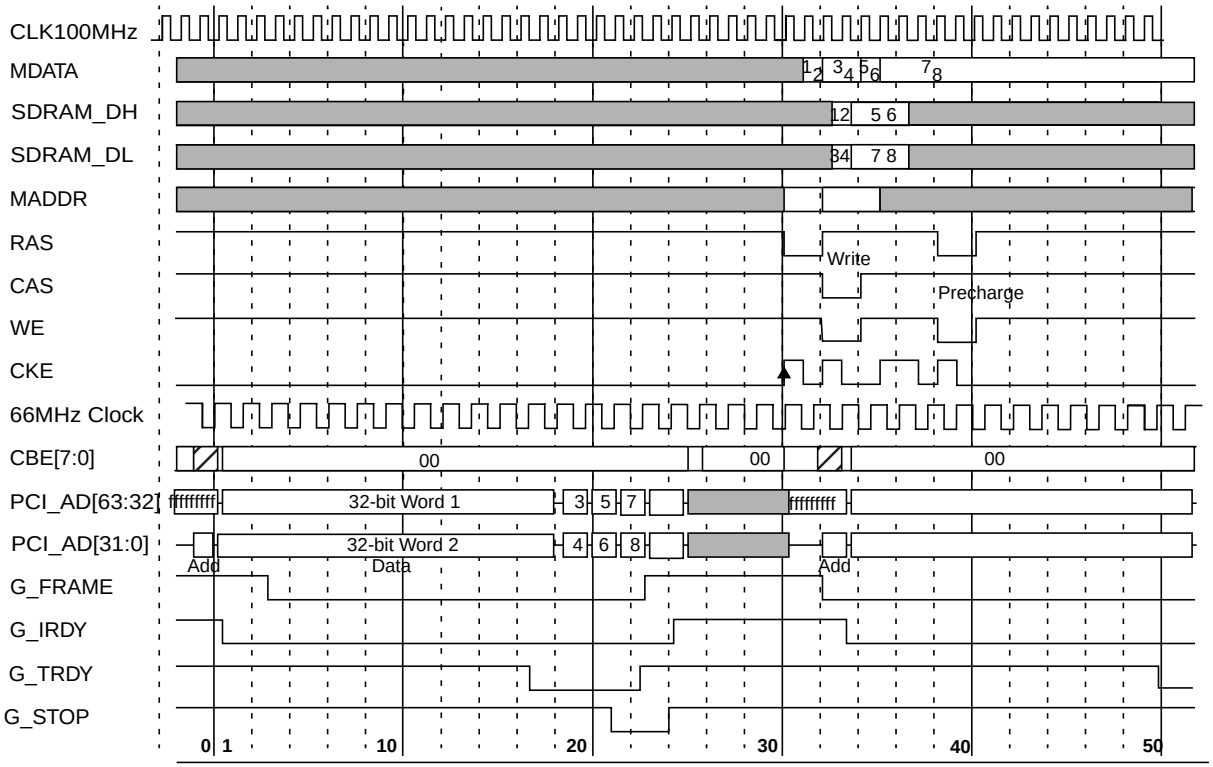


Figure 10-9. Write 32 Bytes to SDRAM from a PCI Master on the 66MHz PCI64 bus

Chapter 11. Signal Summary

Signal Name	Description	I/O
60x Interface		
$\overline{\text{SYS\_BR0:3}}$	Bus Request. Indicates the device on the 60x bus associated with this signal is requesting ownership of the address bus. Should be tied up (1) if unused.	I
$\overline{\text{SYS\_BG0:3}}$	Bus Grant. Indicates the master associated with this signal may, with proper qualification, assume mastership of the address bus.	O
SYS_TS	Transfer Start. **Output:** Indicates that the CPC710 has started an address tenure and the address bus and transfer attribute signals are valid. Only address-only operations and snoop operation with programmable TT code are performed. **Input:** Indicates a master on the 60x has started an address tenure and the address bus and transfer attribute signals are valid. For address tenures that require a data transfer, this signal also indicates a request for the data bus.	I/O
SYS_ADDR00:31	Address Bus. **Output:** Represents the physical address of a cache operation that should be snooped by devices on the 60x bus. A[0] is the most significant address bit. **Input:** Represents the physical address for the current transaction.	I/O
SYS_ADDRP0:3	Address Parity. **Output:** Represents one bit of odd parity for each of the four bytes of the address bus. Odd parity means that an odd number of bits, including the parity bit, are driven high. The signal assignments correspond to the following: AP0 - A0:7 AP1 - A8:15 AP2 - A16:23 AP3 - A24:31 **Input:** Represents one bit of *odd* parity for each of the four bytes of the address bus. A checkstop is generated if bad parity is detected and bit 8 is 1 in the error control register.	I/O
SYS_TT0:4	Transfer Type. **Output:** Indicates the type of transfer in progress. The values are programmable according to the PowerPC type and stored in the CPC0_ATAS register. **Input:** Indicates the type of transfer in progress.	I/O
SYS_TSIZ0:2	Transfer Size. **Output signals and the TBST signal:** Indicate the data transfer size of the operation. The CPC710 sets these signals to a value stored in the CPC0_ATAS register for the operations it initiates. Input signals and the TBST signal: For normal memory accesses, indicate the data transfer size of the operation. For the DMA instructions (eciwx and ecowx), they indicate the 4-bit Resource ID (PCICx_REVID) of the DMA operation ($\overline{\text{TBST}}$ || TSIZ0-TSIZ2).	I/O
SYS_TBST	Transfer Burst. **Output signal and the TSIZ signals:** Indicate the data transfer size of the operation. The CPC710 sets this signal according to the bit in the CPC0_ATAS register for operations it initiates. Input signal: For normal memory accesses, indicates a burst transfer is in progress. For DMA instructions (eciwx and ecowx), the input signal and the TSIZ signals indicate the 4-bit Resource ID (PCICx_REVID) of the DMA operation ($\overline{\text{TBST}}$ || TSIZ0-TSIZ2).	I/O
SYS_GBL	Global. Always asserted by the CPC710 for transactions that it initiates to indicate that all devices on the 60x bus must snoop the transaction. Since the CPC710 asserts this signal only when it is PowerPC bus address master, no contention is possible with PowerPC 750 or 7400 input/output GBL signal connected to $\overline{\text{SYS\_GBL}}$.	O.D
SYS_AACK	Address Acknowledge. Indicates the address tenure is complete and the ARTRY sampling window ends on the following bus cycle. Address bus and transfer attribute signals must go to tri-state on the next bus cycle.	O

Signal Name	Description	I/O
SYS_ARTRY	Address Retry. **Output:** indicates that the CPC710 detects a condition that requires an address tenure to be retried. **Input:** When asserted in response to a CPC710 cache operation, the CPC710 assumes the cache line is modified and/or present in a CPU or L2 cache. The CPC710 then retries the operation on the PCI bus and address tenure is not rerun until the device on the PCI bus reruns its transfer. The pre-charge logic is always signaled to initiate the pre-charge sequence.	I/O
SYS_SHD	Shared. **Output:** Not applicable; The CPC710 only pre-charges the signal. **Input:** Instructs the pre-charge logic to initiate a pre-charge sequence.	I/O
SYS_L2_HIT	L2 Hit. Indicates an external slave has been addressed by the current master. The CPC710 arbiter uses this signal to confirm positive selection of an address tenure on the 60x bus. **Warning:** This signal is subject to timing constraints. Must be tie to V_{DD} = 2.5V when unused.	I
$\overline{\text{SYS\_DBG0:3}}$	Data Bus Grant. Indicates the device associated with this signal may, with the proper qualification, assume mastership of the data bus.	O
SYS_DATA00:63	Data Bus. Byte 0: D[0:7] - DH[0:7] Byte 1: D[8:15] - DH[8:15] Byte 2: D[16:23] - DH[16:23] Byte 3: D[24:31] - DH[24:31] Byte 4: D[32:39] - DL[0:7] Byte 5: D[40:47] - DL[8:15] Byte 6: D[48:55] - DL[16:23] Byte 7: D[56:63] - DL[24:31]	I/O
SYS_DATAP0:7	Data Parity Bus. Represents one bit of odd parity for each of the eight bytes of the data bus. Odd parity means that an odd number of bits, including the parity bit, are driven high. The signal assignments correspond to the following: DP[0]: Data[0:7] DP[4]: Data[32:39] DP[1]: Data[8:15] DP[5]: Data[40:47] DP[2]: Data[16:23] DP[6]: Data[48:55] DP[3]: Data[24:31] DP[7]: Data[56:63]	I/O
$\overline{\text{SYS\_TA}}$	Transfer Acknowledge. **Output:** Indicates a single beat of data transfer between the CPC710 and a master on the 60x bus. For read transfers, indicates the data bus is valid with read data and the master must latch it in. For writes, indicates that the CPC710 has latched in write data from the data bus. The CPC710 asserts the signal for each beat in a burst transfer. **Input:** Indicates a single beat of data transfer has occurred. The CPC710 arbiter uses this signal and the address transfer attribute signals to determine the end of the data bus tenure.	I/O
$\overline{\text{SYS\_TEA}}$	Transfer Error Acknowledge. **Output:** Indicates that the CPC710 has detected an error condition and that a machine check exception is desired. Assertion of this signal terminates the current data bus tenure. The CPC710 can be set up to transform any $\overline{\text{SYS\_TEA}}$ to normal $\overline{\text{SYS\_TA}}$ with machine check condition signaling on $\overline{\text{SYS\_MCP0}}$, $\overline{\text{SYS\_MCP1}}$, $\overline{\text{SYS\_MCP2}}$, or $\overline{\text{SYS\_MCP3}}$. **Input:** Informs the CPC710 60x bus arbiter that the current data bus tenure has been terminated.	I/O
$\overline{\text{SYS\_MCP0:3}}$	Machine Check. Indicates that the CPC710 has detected an error condition and a machine check exception is desired. External pull-up is required.	3 state
$\overline{\text{CHKSTOP}}$	Checkstop. Indicates that the CPC710 has detected a non-recoverable error condition and has entered checkstop state.	O.D

Signal Name	Description	I/O
SYS_HRESET0:3	Hard Reset. Indicates the device or card associated with this signal must initiate a complete hard reset. All outputs should be released to tri-state. Duration of reset, except for device hardware system reset, is controlled by software.	O
SYS_SRESET0:3	Soft Reset. Indicates the processor connected to this signal will take a reset exception. Occurs following a write to the CPU soft reset register (CPC0_SRST) that has the appropriate bit set.	O
SYS_TBE	Timebase Enable. Indicates the processor time bases should continue counting. Reflects bit 12 of the CPC0_UCTL12 register 0xFF001000.	O
POWERGOOD	Normal operation when up (1) General system reset when down (0)	I
IT1	Interrupt 1. Generated after writing 1 in the PCIC1_ITADDSET interrupt register. This interrupt can be used by an external interrupt controller. The writing can be made from the CPU in configuration mode or from the PCI64 bus. Only the PowerPC CPU can reset the interrupt by writing 1 in the PCIL1_ITADDRESET interrupt reset register.	O
IT2	Interrupt 2. Indicates the end of the DMA data transfer. Corresponds to assertion of bit 4 in the DMA0_GSCRx status register.	O
GPIO0:2	General purpose I/O signals.	I/O
DLK	Programmed by setting bit 20 of the CPC0_PGCHP register: 0: DLK (see below) Deadlock. Asserted when processor range of address is out of the non-deadlock zone. An address SYS_ARTRY is sent to the PowerPC when DLK is set.	O
NODLK	Programmed by setting bit 20 of the CPC0_PGCHP register: 0: NODLK (see below) Deadlock Disable. Used only when the deadlock address range checking is programmed: Asserted (0), deadlock checking is disabled If tied high (1), deadlock checking can be performed	I
SYS_TA_HIT	External Transfer Acknowledge Hit. A transition from high to low of this signal results in the generation of the SYS_TA output signal in the following system clock cycle. Must be tied to V_{DD} =3.3V when unused.	I
SDRAM Interface		
BS1:0	Internal Bank Select.	O
MDATA00:63	Memory Data.	I/O
MDATA64:71	Memory Data ECC bits.	I/O
MADDR0_ODD	Memory Address bit 0 for odd DIMMs.	O
MADDR0_EVEN	Memory Address bit 0 for even DIMMs.	O
MADDR13:1	Memory Address bits 13 to 1 (13 is msb).	O
SDCS00:11	SDRAM Chip Select.	O
SDCKE0:9	SDRAM Clock Enable. Ten signals with same shape for buffering issues.	O
WE0:1	Memory Write Enable (two signals with same shape for buffering issues). WE1 can be converted in a Chip Data Mask (SDDQM) by setting bit 14 of the SDRAM0_MCCR register.	O
SDRAS0:1	SDRAM Row Address Strobe (two signals with same shape for buffering issues). SDRAS1 can be converted in a Chip Data Mask (SDDQM) by setting bit 14 of the SDRAM0_MCCR register.	O
SDCAS0:1	SDRAM Column Address Strobe (two signals with same shape for buffering issues). SDCAS1 can be converted in a Chip Data Mask (SDDQM) by setting bit 14 of the SDRAM0_MCCR register.	O

Signal Name	Description	I/O
SDDQM	Data Output Mask: same shape signal available on I/Os WE1, SDRAS1, SDCAS1, or G_ARB after setting bits 14:15 of SDRAM0_MCCR register.	O
MUX_OEA:B	Output Enable of Data to Port A or B.	O
MUX_CLKEN1B MUX_CLKEN2B	Clock Enable of Data sent to the Memory (two signals with same shape for buffering issues).	O
MUX_CLKENA1 MUX_CLKENA2	Clock Enable of Data sent to the CPC710. On Clock A1 the first part of the data is stored in the external MUX controller, and on clock A2 full transfer is done.	O
MUX_SEL	Control the MUX circuit of the external MUX controller.	O
PCI32 Interface		
PCI_CLK	Main clock input for the PCI32 bit bridge (maximum 33 MHz).	I
P_ADL31:00	32-bit Multiplexed Address/Data. A write operation is defined as the transfer of data from the PCI bus master to a PCI slave device on the PCI Bus.	I/O
P_CBE3:0	Bus Command/Byte Enable.	I/O
P_DEVSEL	Device Select.	I/O
P_FRAME	Cycle Frame. Driven by the current master to indicate the beginning and duration of an access.	I/O
P_IRDY	Initiator Ready.	I/O
P_LOCK	Lock. Used to establish, maintain, and release resource locks on PCI32. Reserved for future use. Tying up this signal is recommended.	I
P_MEMACK	Memory Acknowledge. Indicates that the CPC710 has flushed all CPU to PCI32 bus buffers and any CPU access to PCI is being SYS_ARTRYed.	O
P_MEMREQ	Memory Request. Indicates a PCI device accessing system memory has a potential deadlock and requests the CPC710 to flush all posted CPU to PCI buffers and ARTRY all PCI32 bus transfers from the 60x bus.	I
P_PAR	Parity Bit.	I/O
P_GNT0:3 [P_GNT4:6]G_GNT5:7	PCI32 Bus Grants.	O
P_REQ0:3 [P_REQ4:6]G_REQ5:7	PCI32 Bus Requests. P_REQ2 is sampled at Reset, to select arbitration on the PCI-32 bus. The arbitration can be made by the CPC710 (P_REQ2 = 1) or by external circuit (P_REQ2 = 0). In case of external arbitration, the request is send to PCI from P_GNT1 and the grant from the external arbiter is received on pin P_REQ1. Extended Flash is available only when the CPC710 is the PCI32 arbiter.	I
P_RST	PCI32 Bus Reset.	O
P_PERR	PCI32 Data Parity Error.	I/O
P_SERR	PCI32 System Parity Error. Reports parity errors on address, special cycle data, or systems.	I/O
P_STOP	Stop. Asserted by the target to request the master to stop current transaction.	I/O
P_TRDY	Target Ready. Asserted by the target when ready to receive data.	I/O
PCI64 Interface		
PCG_CLK	Main clock input for the PCI64 bit bridge (maximum 66 MHz).	I

Signal Name	Description	I/O
G_ADH31:00	32-bit Multiplexed Address/Data Higher Part. In the address phase when G_REQ64 is asserted, these bits are the upper part of 64 bit address AD63:32. During data phase an additional 32-bits of data are transferred when G_REQ64 and G_ACK64 are both asserted.	I/O
G_ADL31:00	32-bit Multiplexed Address/Data Lower Part. A write operation is defined as the transfer of data from the PCI bus master to a PCI slave device on the PCI Bus.	I/O
G_ACK64	Acknowledge 64-bit transfer.	I/O
G_REQ64	Request 64-bit transfer. External pull-up required	I/O
G_PAR64	Parity upper double word.	I/O
G_CBE7:0	Bus Command/Byte Enable.	I/O
G_DEVSEL	Device Select.	I/O
G_IDSEL	Initialization Device Select. Used as chip select during configuration.	I
G_FRAME	Cycle Frame	I/O
G_IRDY	Initiator Ready	I/O
G_LOCK	Lock. Used to establish, maintain and release resource locks on PCI64. Reserved for future usage. It is recommended to tie up this signal.	I
G_PAR	Parity bit.	I/O
G_GNT0:4 G_GNT5:7[P_GNT4:6]	Bus Grants.	O
G_REQ0:4 G_REQ5:7[P_REQ4:6]	Bus Requests. G_REQ[2] is sampled at Reset, to select arbitration on the PCI64 bus. The arbitration can be made by the CPC710 (G_REQ2 = 1) or by external circuit (G_REQ2 = 0). In case of external arbitration, the request is send to PCI from G_GNT1 and the grant from the external arbiter is received on pin G_REQ1. G_REQ5:7 are programmed by setting bits 21:23 of the CPC0_PGCHP register	I
G_RST	Reset PCI64 Bus. External pull-up required. **Input:** Replicated on G_RESETOUT when programmed (no internal use). **Output:** Activated by the CPC710 at power up or by programming.	I/O
G_PERR	Data Parity Error.	I/O
G_SERR	System Parity Error.	I/O
G_STOP	Stop. Asserted by the target to request the master to stop the current transaction.	I/O
G_TRDY	Target Ready. Asserted by the target when ready to receive data.	I/O
G_INTA:D	Interrupts A:D.	O
G_ARB	Arbiter. Asserted when the CPC710 is the PCI64 arbiter. Can be converted in a Chip Data Mask (SDDQM) by setting bit 15 of the SDRAM0_MCCR registe.r	O
G_RESETOUT	Local Reset. Asserted by PCI64 reset and special conditions.	O.D
SIO Interface		
FLASH_CE	Extended Flash Chip Enable. This signal goes to 0 after the CPC710 has decoded an access to the Extended Flash address range. 1: Boot Flash Enabled. 0: Extended Flash Enable. This signal must be used on card to insure that Boot Flash and Extended Flash cannot be accessed at the same time.	O
FLASH_OE	Output Enable. Flash ROM.	O
FLASH_WE	Write Enable. Flash ROM.	O
PRES_OE0:1	Output Enable. Presence detect (PD) buffer 0 or buffer 1.	O
XADR_LAT	Latch Signal. For SIO address register.	O
XCVR_RD	Address Direction. SIO address bus.	O

Signal Name	Description	I/O
JTAG Interface		
TDI	Test Data In.	I
TMS	Test Mode Select.	I
TDO	Test Data Out.	O
TCK	Test Clock.	I
$\overline{\text{TRST}}$	Reset.	I
System Interface		
SYS_CLK	System Reference Clock. Used as: 1. 60X bus clock 2. Attached proccesor clock 3. Synchronous SDRAM signals This clock is not synchronized with the PCI-32 and the PCI64 clocks.	I
PLL_RANGE1:0	PLL frequency range selector for the System Clock. 00: 50 to 100 MHz 01: 58 to 114 MHz 10: 66 to 134 MHz 11: 80 to 160 MHz	I
PLL_LOCK	Output indicating the PLL is locked.	O
PLL_TUNE5:0	Loop stability tuning control of the PLL . Recommended values: 010101 if range is 50 to 100 MHz 010011 if range is 58 to 114 MHz 010011 if range is 66 to 134 MHz 010011 if range is 80 to 160 MHz	I
$\overline{\text{PLL\_RESET}}$	Reset and Bypass mode enable of the PLL	I
$\overline{\text{CE0\_TEST}}$	Reserved	I
Power		
GND	Ground.	n/a
OV_{DD}	Output driver voltage—3.3V.	n/a
V_{DD}	Logic voltage—2.5V.	n/a
AV_{DD}	Analog voltage—2.5V. Filtered supply for PLL circuits.	n/a
Other pins		
Reserved	Do not connect signals, voltage, or ground to these pins.	n/a

Index

C

CPC0_ABCNTL 9-10
CPC0_ATAS 9-13
CPC0_AVDG 9-15
CPC0_ERRC 9-17
CPC0_GPDIR 9-19
CPC0_GPIN 9-20
CPC0_GPOUT 9-21
CPC0_MPSR 9-22
CPC0_PCIBAR 9-23
CPC0_PCICNFR 9-24
CPC0_PCIENB 9-25
CPC0_PGCHP 9-26
CPC0_PIDR 9-29
CPC0_RGBAN0 9-30
CPC0_RGBAN1 9-31
CPC0_RSTR 9-32
CPC0_RTBR 9-33
CPC0_SEAR 9-34
CPC0_SESR 9-35
CPC0_SIOC0 9-38
CPC0_SIOC1 9-40
CPC0_SPOR 9-41
CPC0_SRST 9-42

D

DMA0_GSCRx 9-45
DMA0_UXWAR 9-53
DMA0_XCLRx 9-47
DMA0_XPARx 9-48
DMA0_XSCRx 9-49
DMA0_XSSRx 9-50
DMA0_XTARx 9-52

P

PCIC0_DLKRETRY 9-54
PCIC1_INTRESET 9-55
PCIC1_ITADDSET 9-56
PCIC1_PPBAR 9-57
PCIC1_PSBAR 9-58
PCICx_BIST 9-59
PCICx_BUSNO 9-60
PCICx_CACHELS 9-61
PCICx_CLS 9-62
PCICx_CMD 9-63
PCICx_HDTYPE 9-67
PCICx_INTLN 9-68
PCICx_INTPN 9-69
PCICx_LATTIM 9-70
PCICx_MAXLTNCY 9-71
PCICx_MINGNT 9-72
PCICx_REVID 9-74
PCICx_STATUS 9-75
PCICx_SUBNO 9-77
PCICx_VENDID 9-78
PCIL0_PPBAR 9-79
PCIL0_PSBAR 9-80
PCIL1_ITADDRESET 9-82
PCILx_BARPS 9-85
PCILx_BIODLK 9-86
PCILx_CFGADDR 9-88
PCILx_CFGDATA 9-89
PCILx_CRR 9-90
PCILx_CRTLW 9-92
PCILx_DLKCTRL 9-93
PCILx_DLKDEV 9-95
PCILx_INTACK 9-96
PCILx_IOSIZE 9-97
PCILx_MSIZE 9-98
PCILx_PCIDG 9-99
PCILx_PIBAR 9-100
PCILx_PLSSR 9-101
PCILx_PMBAR 9-102
PCILx_PPSIZE 9-103
PCILx_PR 9-104
PCILx_PSEA 9-106
PCILx_PSRCR 9-107
PCILx_PSSIZE 9-109
PCILx_PSWCR 9-110
PCILx_SIBAR 9-112
PCILx_SMBAR 9-113
PCILx_TIODLK 9-114
PCILx_TPMDLK 9-115

R

registers
- CPC0_ABCNTL 9-10
- CPC0_ATAS 9-13
- CPC0_AVDG 9-15
- CPC0_ERRC 9-17
- CPC0_GPDIR 9-19
- CPC0_GPIN 9-20
- CPC0_GPOUT 9-21
- CPC0_MPSR 9-22
- CPC0_PCIBAR 9-23
- CPC0_PCICNFR 9-24
- CPC0_PCIENB 9-25
- CPC0_PGCHP 9-26
- CPC0_PIDR 9-29
- CPC0_RGBAN0 9-30
- CPC0_RGBAN1 9-31
- CPC0_RSTR 9-32
- CPC0_RTBR 9-33
- CPC0_SEAR 9-34
- CPC0_SESR 9-35
- CPC0_SIOC0 9-38
- CPC0_SIOC1 9-40
- CPC0_SPOR 9-41
- CPC0_SRST 9-42
- DMA0_GSCRx 9-45
- DMA0_XCLRx 9-47
- DMA0_XPARx 9-48
- DMA0_XSCRx 9-49
- DMA0_XSSRx 9-50

DMA0_XTARx 9-52
DMA0_XWARx 9-53
PCIC0_DLKRETRY 9-54
PCIC1_INTRESET 9-55
PCIC1_ITADDSET 9-56
PCIC1_PPBAR 9-57
PCIC1_PSBAR 9-58
PCICx_BIST 9-59
PCICx_BUSNO 9-60
PCICx_CACHELS 9-61
PCICx_CLS 9-62
PCICx_CMD 9-63
PCICx_HDTYPE 9-67
PCICx_INTLN 9-68
PCICx_INTPN 9-69
PCICx_LATTIM 9-70
PCICx_MAXLTNCY 9-71
PCICx_MINGNT 9-72
PCICx_REVID 9-74
PCICx_STATUS 9-75
PCICx_SUBNO 9-77
PCICx_VENDID 9-78
PCIL0_PPBAR 9-79
PCIL0_PSBAR 9-80
PCIL1_ITADDRESET 9-82
PCILx_BARPS 9-85
PCILx_BIODLK 9-86
PCILx_CFGADDR 9-88
PCILx_CFGDATA 9-89
PCILx_CRR 9-90
PCILx_CTRLW 9-92
PCILx_DLKCTRL 9-93
PCILx_DLKDEV 9-95
PCILx_INTACK 9-96
PCILx_IOSIZE 9-97
PCILx_MSIZE 9-98
PCILx_PCIDG 9-99
PCILx_PIBAR 9-100
PCILx_PLSSR 9-101
PCILx_PMBAR 9-102
PCILx_PPSIZE 9-103
PCILx_PR 9-104
PCILx_PSEA 9-106
PCILx_PSRCR 9-107
PCILx_PSSIZE 9-109
PCILx_PSWCR 9-110
PCILx_SIBAR 9-112
PCILx_SMBAR 9-113
PCILx_TIODLK 9-114
PCILx_TPMDLK 9-115
SDRAM0_MCER0:5 9-120
SDRAM0_MEAR 9-123
SDRAM0_MWPR 9-125
SDRAM0_SIOR0 9-127
SDRAM0_SIOR1 9-128

S

SDRAM0_MCER0:5 9-120
SDRAM0_MEAR 9-123
SDRAM0_MWPR 9-125
SDRAM0_SIOR0 9-127
SDRAM0_SIOR1 9-128

Revision Log

Revision Summary for Fourth Edition, September 2002

Chapter	Description
4	4.6 Deadlock Avoidance. Most of this section (two pages) replaced by new content.
8	8.1 and 8.3. $\overline{\text{INT2}}$ signal name changed to $\overline{\text{IT2}}$, as in CPC710 datasheet.
9	Table 9-3, renamed Deadlock Control register to Deadlock Avoidance Control register Edited CPC0_PGCHP register Edited CPC0_SESR register Edited CPC0_SIOC0 register Edited DMA0_GSCRx register Edited DMA0_XSSRx register Edited PCILx_DLKDEV register Edited PCILx_PCIDG register Edited PCILx_PR register Edited PCILx_PSWCR register Edited SDRAM0_MCCR register

© Copyright International Business Machines Corporation 2002

All Rights Reserved

IBM and the IBM logo are registered trademarks of the IBM Corporation.

All information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in implantation, life support, space, nuclear, or military applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change IBM product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN “AS IS” BASIS. In no event will IBM be liable for any damages arising directly or indirectly from any use of the information contained in this document.

IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6531

The IBM home page can be found at
http://www.ibm.com

The IBM Microelectronics Division home page can be found at
http://www.chips.ibm.com

Document No. SA14-2571-02